2023

Deep Video Understanding with Model Efficiency and Sparse Active Labeling

Aayush Jung Bahadur Rana
University of Central Florida

DEEP VIDEO UNDERSTANDING WITH MODEL EFFICIENCY AND SPARSE ACTIVE LABELING

by

AAYUSH JUNG BAHADUR RANA
M.S. University of Central Florida, 2023
B.Sc.E. Asian Institute of Technology, 2015

A dissertation submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy
in the Department of Computer Science
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Summer Term
2023

Major Professor: Yogesh Singh Rawat

ABSTRACT

Videos capture the inherently sequential nature of the real world, making automatic video understanding an essential need for automatic understanding of the real world. Due to major advancements in camera, communication, and storage hardware, videos have become a widely used data format for crucial applications such as home automation, security, analysis, robotics, and autonomous driving. Existing methods for video understanding require heavy computation and large training data for good performance, this limits how quick the videos can be processed and how much data can be labeled for training. Real-world video understanding requires analyzing dense scenes and sequential information, which increases the processing time and labeling cost as the video increases in scene density and video length. Therefore, it is crucial to develop video understanding methods that reduces the processing time and labeling cost.

In this dissertation, we first propose a method to improve network efficiency for video understanding task and then provide methods to improve annotation efficiency for video understanding task. Through these works, we aim to improve the network efficiency as well as data annotation efficiency, as an effort to encourage wider development and adaptation of large scale video understanding methods. First, we propose an end-to-end neural network which performs faster video actor-action detection. Our proposed network reduces the need for extra region proposal computation and post-process filter, making the network training easy as well as increasing the inference speed. Next, we propose an active learning based sparse labeling method that makes large video dataset annotation efficient. It selects a few useful frames for annotation from videos, reducing annotation cost while maintaining the dataset usefulness for video understanding task. We also provide a method to train existing video understanding models using such sparse annotations. Then, we propose a clustering-based hybrid active learning method that also selects useful videos along with useful frames for annotation, reducing annotation cost even further. Finally, we study

the relation between different types of annotations and how they impact video understanding task. We extensively evaluate and analyze our methods on various dataset and downstream tasks to show that they can do efficient video understanding with faster network and limited sparse annotations.

EXTENDED ABSTRACT

Dense video understanding is a challenging problem with a wide range of applications in automation, analysis, security, autonomous driving and robotics. Real-world videos have varying object density due to crowded scene and they have varying video length based on the underlying application. This creates challenge for video understanding as the processing time increases with video length and density. Better video understanding methods require more annotated data for training, which also gets costlier to label as the video gets longer and more crowded. These challenges need to be addressed to overcome different video understanding task such as classification, detection, segmentation, tracking, summarization and more. As complex task such as detection and segmentation need more processing time and labeled data, it is important to time and label efficiency for dense video understanding.

While deep neural networks for image understanding have improved a lot over the years, it is not the same for video understanding task. This comes down to two factors: video models have extra temporal information which increases the computation cost of the model during training and inference, and video datasets with full annotations are costly to label. To improve video understanding further, we need to optimize the models for efficient training and inference by streamlining the entire process. Furthermore, we need to design methods that enable labeling large video dataset efficiently by reducing unnecessary annotation cost, along with methods that enable training video understanding models using sparse annotations.

In chapter 3, we propose a simple yet effective end-to-end deep network for actor-action detection in videos. The existing methods take a top-down approach based on region-proposal networks (RPN), where the action is predicted based on the detected proposals followed by post-processing such as non-maximal suppression. While effective in terms of performance, these methods pose

limitations in scalability for dense video scenes with a high memory requirement for thousands of proposals. We propose to solve this problem from a different perspective where we don't need any proposals. We propose a method to perform pixel level joint actor-action detection, where every pixel of the detected actor is assigned an action label. The proposed method is a fully convolutional network which does not require any proposals and post-processing making it memory as well as time efficient. This also allows it to be scalable to dense video scenes as its memory requirement is independent of the number of actors present in the scene. The network is trained end-to-end with actor and action detection loss and performs competitively to prior works on video actor-action detection task, while having less computation time and memory requirements.

In chapter 4, we focus on efficient labeling of videos for action detection to minimize the annotation cost for video action detection training tasks. We propose a novel active learning strategy for sparse labeling for video action detection. Sparse labeling will reduce the annotation cost but poses two main challenges: 1) how to estimate the utility of annotating a single frame for action detection as detection is performed at video level?, and 2) how these sparse labels can be used for action detection which require annotations on all the frames? We address these challenges within a simple active learning framework. For the first challenge, we analyze frame-level scoring mechanisms aimed at selecting the most informative frames for action detection and propose an intuitive mechanism for video action detection. Next, we introduce a novel loss formulation which enables training of action detection model with these sparsely selected frames. We train the network end-to-end using this loss formulation for sparse labels and use proposed active learning framework to select high utility frames for further annotation. We demonstrate that the proposed approach outperforms all existing baselines, with performance comparable to fully-supervised approach.

In chapter 5, we show an active learning based framework that can reduce annotation cost of frames from all videos which improves video action detection. In chapter 5 we focus on further reducing annotation cost by also selecting videos for annotation which will have more impact on video

action detection training. We present a hybrid active learning strategy which performs efficient labeling by first selecting videos (inter-sample) for labeling and then selecting a few frames from these videos (intra-sample) to be annotated. This strategy reduces the annotation cost from two different aspects leading to significant labeling cost reduction. We use video level and frame level informativeness and diversity to select useful samples while reducing redundancy. We train the network end-to-end with sparse labels using improved loss formulation from chapter 4 and select samples using proposed method to reduce annotation cost while performing competitively with prior works.

In previous chapters we show how to improve network efficiency and how to reduce annotation cost for video action detection. These methods work on the same type of annotation for the entire dataset. In chapter 6, we analyze the types of annotation appropriate for each sample and how it affects video action detection. We focus on two different aspects affecting video action detection: 1) how to obtain varying levels of annotations for videos, and 2) how to learn video action detection with different types of annotations. We study several annotation types including i) video level tags, ii) points iii) scribbles, iv) bounding box, and v) pixel level masks. We propose a simple active learning strategy which estimates appropriate types of annotations needed for each video sample based on the usefulness of that video sample. We also propose a spatio-temporal deep superpixel method which generates pseudo-labels from different types of annotations and enables learning of video action detection from such annotations.

ACKNOWLEDGMENTS

I would like to express sincere gratitude to my Ph.D. advisor Dr. Yogesh Singh Rawat for his guidance and support throughout my doctoral journey. I am also grateful to Dr. Mubarak Shah for his invaluable mentorship during the early years of my doctoral studies and for his feedback as my committee member. Also, I would like to extend my gratitude to my committee members Dr. Chen Chen and Dr. Shaurya Agarwal for their precious time and valuable comments on my research dissertation.

Additionally, I would like to extend my appreciation to all past and present members of the CRCV family whom I was fortunate to cross paths with, Cherry Place, Tonya LaPraire, Dr. Abhijit Mahalanobis, Dr. Niels Da Vitoria Lobo, Dr. Ulas Bagci, Dr. Shruti Vyas, Dr. Eyasu Mequaanint, Dr. Leulseged Alemu, Dr. Yonatan Tesfaye, Dr. Navid Kardan, Dr. Gaurav Nayak, Dr. Aidean Sharghi, Dr. Aisha Urooj Khan, Dr. Amir Mazaheri, Dr. Bruce McIntosh, Dr. Gonzalo Vaca, Dr. Kevin Duarte, Dr. Khurram Soomro, Dr. Krishna Regmi, Dr. Mahdi M. Kalayeh, Dr. Marzieh Edraki, Dr. Nasim Souly, Dr. Naji Khosravan, Dr. Rodney Lalonde, Aakash Kumar, Adeel Yousaf, Akash Kumar, Ishan Dave, Jyoti Kini, Madeline Chantry Schiappa, Mahfuz Al-Hasan, Mamshad Nayeem Rizve, Manu Pillai, Muhammad Tayyab, Nyle Siddiqui, Parth Parag Kulkarni, Priyank Pathak, Rajat Modi, Robert Browning, Rohit Gupta, Sarinda Samarasinghe, Swetha Sirnam, Tushar Sangam, Ankit Sharma, Babak Ebrahimi, Furkan Cimen and Ugur Demir for their support, friendship and fond memories. I would also like to thank my friends Sunny Amatya, Su Soe, Antonio P Jegillos Jr., Sarawut Yaroh, Hadia Kanokkorn, and Bon-z Soe for the love and support through all these years.

Lastly, to my parents, sibling, grandparents, and cousins, I am forever grateful for your unconditional support and continuous encouragement for all my goals throughout life.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

Recent advancements in deep learning have made computer vision widely used in many real-world tasks involving videos, images and texts. Generally, video related tasks have higher computation requirements and lower data availability compared to image and text based tasks, resulting in slower adaptation and development of video specific methods along with an increased challenge from noise and perturbations [11, 12]. Video understanding is an important task with growing demand in analysis, security, autonomous driving and robotics [13, 14, 15, 16, 17, 18]. Since videos have spatial as well as temporal information which is required to solve the underlying task such as classification, temporal localization, detection, segmentation and summarization, simply extending image based solutions does not give the best results in videos [19, 6, 20, 4]. As shown in Figure 1.1, different video dataset have different annotation types and focus on different underlying task which makes video understanding challenging. Our aim is to improve the computation efficiency of the network for video understanding tasks and to reduce the cost for annotating large video dataset effectively.

Figure 1.1: Samples for various video understanding tasks for different dataset. 1^{st} column shows video actor-action detection with pixel-wise annotation for A2D dataset [1]. 2^{nd} column shows video action detection with pixel-wise annotation for JHMDB-21 dataset [2]. 3^{rd} column shows video action detection with bounding box annotation for UCF-101-24 dataset [3]. 4^{th} column shows video object segmentation with pixel-wise annotation for YouTube-VOS dataset [4].

We focus on detection and segmentation tasks for improving video understanding. First, we look into improving the network efficiency for video actor-action detection while improving performance. Video actor-action task requires detecting the actors in a given scene and then classifying all the actions each actor performs. While actor detection can be done solely from single image, the associated actions will need temporal information as they can be ambiguous in a single frame (car-run vs car-stop, person-walk vs person-stand). Existing works expand on image based approach on videos, which relies on a costly per-frame region proposal and pooling algorithm [21, 22]. We propose a simple end-to-end method without using any region proposal, which performs competitively with prior works while reducing the computation time.

Next, we focus on improving dataset labeling efficiency for video action detection task. Video action detection has been a challenging task as it requires the localization of where an action occurs spatially within a frame as well as when it occurs temporally across multiple frames [23, 3, 24, 2, 5, 25]. Recent advances in video action detection has stemmed from large models trained on fully annotated data with bounding box and pixel-level masks [24, 26, 27]. Unlike video action classification task which can be trained with only class labels for the video [6, 19], action detection training requires most of the frames to be correctly annotated. This poses a problem for scaling to large datasets as the number of frames to annotate increases rapidly, limiting big datasets to subsample frames for annotation [28, 29]. Lack of large dataset availability also limits the improvement in video action detection task. Therefore, it is important to improve video action detection training using low-cost annotation alternatives. We propose active learning based frame and video selection algorithms, which identifies and picks high utility samples for human annotation and reduces redundant and noisy annotation. We develop our algorithm to work on frame-level and expand it to work on video-level, reducing annotation cost for video dataset while still maintaining comparative action detection performance. Finally, we look into different types of low-cost annotations which can be used for dataset annotation while maintaining the performance

goals.

In the following sections, we first introduce our method for making an efficient network to perform video actor-action detection. We then briefly describe our methods to reduce annotation cost associated with video action detection training. Finally, we explain various types of low-cost annotations and their relationship with video action detection training.

1.1 Efficient Actor-Action Detection in Videos

Actor-action detection in videos is a challenging problem where the goal is to detect all the actors in the video and determine which different actions they are performing. Previous methods use per frame object detection using region proposal and pooling method [30, 31] to identify all actors and classify their actions [21, 22]. With multiple actors and multiple actions per actor in a video as shown in Figure 1.2, these methods become complex and inefficient as thousands of region proposals per frame are required, followed by a post-processing cleanup using non-maximal suppression. Due to these limitations, such networks have to be trained in multiple stages [21], leading to an increase in the training and inference time.

In this work, we propose a streamlined end-to-end network which does not require any proposals and performs single shot actor-action detection in videos. We use an encoder-decoder based unified network, which leverages spatio-temporal contextual information between objects and their surrounding pixels for joint detection of multiple objects and activities in multiple input video frames at once. Contrary to region proposal based work, we propose an attention based masking approach which adds emphasis on spatio-temporal features belonging to all actors and their surroundings to make better action prediction. This attention masking is applied to the entire video in a single step which makes the proposed approach much more efficient for dense video scenes.

Figure 1.2: Samples from A2D dataset showing the various actor-action pairs. The color of the label corresponds to the actor-action pair highlighted in the image.

We show through extensive evaluation in A2D and VidOR dataset that the proposed network is effective and comparative with prior works while being the fastest method.

1.2 Video Action Detection Using Sparse Labels

Action detection in videos is challenging due to increased computation from temporal data and lack of large scale densely annotated dataset. In previous work, we showed that computation can be reduced to predict actor-action pairs in videos while maintaining performance. Next, we deal with the issue of lack of densely annotated dataset. Unlike image domain, video datasets are more costlier to annotate to provide large data variation and samples as shown in Figure 1.3 and 1.4. Prior semi-supervised methods using partial labels suffer from performance loss while saving annotation cost. To deal with high annotation cost, we propose an active learning based sparse labeling method

which identifies high utility frames for annotation. This approach reduces redundant information from frames and selects fewer frames from videos, reducing cost of annotation required to perform competitively to fully supervised methods.

Figure 1.3: Samples from UCF-101-24 dataset for action detection with bounding-box annotation. Each sample has consecutive action frames with bounding box annotation per action instance.

In this work, we propose an uncertainty-based frame scoring mechanism for videos, termed Adaptive Proximity-aware Uncertainty (APU). APU estimates the frame's utility using the uncertainty of the detections and its proximity from existing annotations, determining diverse set of *informative* frames in a video which are more effective for learning the task of action detection. In addition, we propose a loss formulation which uses weighted pseudo-labeling for effective learning from sparsely labeled videos. Together, the proposed cost estimation algorithm based on APU and the loss function helps in reducing the annotation cost while improving model performance at the same time. We evaluate the proposed methods in UCF-101-24, J-HMDB-21 and YouTube-VOS for video action detection and segmentation tasks.

Figure 1.4: Samples from JHMDB-21 dataset for action detection with pixel-wise mask annotation. Each sample has consecutive frames annotated with pixel-wise mask for each action instance.

1.3 Video Action Detection Using Hybrid Active Learning

As we have shown in earlier work, the annotation cost for video action detection can be reduced using the proposed utility estimation method. While effective, that estimation method only works on frame level by selecting the most useful frame per video. We can further improve the cost reduction by removing less useful and redundant videos entirely, and only selecting few frames from high utility videos. To this end, we propose a hybrid active learning approach that uses clustering to select videos to sample frames from. We propose intra-sample selection which targets informative frames within a video and inter-sample selection which aims at informative samples at video-level.

In this work, we propose Clustering-Aware Uncertainty Scoring (CLAUS), a clustering assisted

active learning strategy which considers informativeness and diversity for sample selection. It relies on model uncertainty for informative sample selection and clustering based selection criteria for reducing redundancy. To use the annotated sparse frames effectively, we improve on prior loss formulation to consider temporal continuity for pseudo-label weights. We perform extensive evaluation of the proposed approach on UCF-101-24 and JHMDB-21 dataset and demonstrate that it performs comparatively with prior fully-supervised works.

1.4 Video Action Detection Using Varying Level of Supervision

In previous works, we have shown that video action detection can be improved on inference time using efficient networks and on annotation cost using active learning based selection strategies. These still require all the annotation to be of the same type for the entire dataset, which is either a bounding-box or pixel-wise mask. However, there are different types of annotation which can also be used for video understanding task with trade-off in performance. To this end, we study how varying types of low to high cost annotations can be used for video action detection to further reduce the labeling cost while maintaining performance. Our goal is to be able to train a good video action detection model using less annotation cost compared to prior weakly and semi-supervised methods. In this work, we propose a simple active learning strategy to estimate which type of annotation is sufficient for each video. We also propose pseudo-label generation method which enables using these different types of annotations together to train a video action detection network.

1.5 Organization

In Chapter 2, we present existing literature for video action detection methods and limited label learning works. In Chapter 3, we present an efficient video actor-action detection network. In

Chapter 4, we present an active learning strategy to reduce annotation cost in videos. In Chapter 5, we present a hybrid active learning method that reduces both video and frame annotation cost and expands the previous chapter further. In Chapter 6, we present a study on annotation types, cost and their effect on network performance and propose a method to select correct type of annotation to reduce overall annotation cost.

CHAPTER 2: LITERATURE REVIEW

In this chapter we provide reviews for prior works pertaining to this dissertation. In Section 2.1 we provide literature pertaining action detection in videos. Section 2.2 contains methods for actor-action detection in videos. Section 2.3 explains existing works on weakly and semi-supervised learning, followed by Section 2.4 containing methods for video action detection using limited labels. Section 2.5 contains literature using active learning for limited label learning and Section 2.6 contains methods using varying level of supervision for training.

2.1 Action detection in videos

Action detection in videos require spatio-temporal localizations of actors in each frame which is then used for classification. Extending the image classification models [30, 32, 33, 34, 35], prior CNN based works detect actors in each frame and combine them temporally to form action tubes while classifying at clip level [36, 37, 24, 38], leveraging existing classification techniques from [6, 19, 23, 39, 40, 41, 42, 43]. While some prior works use a separate region proposal network to detect potential actors [23, 44], using a complicated two-stage process, other prior works have proposed simplified single-stage approach [5, 45]. [23] uses RPN based approach to detect actors in each frame and then forms action tubes by stitching them together, followed by Tube of Interest (TOI) pooling and action classification. [20] does TOI pooling based on foreground segmentation map from an encoder-decoder based network. [44] uses RPN along with transformer based attention map that detects and classifies actions. [5] uses a 3D capsule based CNN, where the authors apply routing-by-agreement algorithm to capture various action representations, leading to localize actions and classify them at the same time. Although prior works show great improvements on action detection in videos, they are limited by complex region proposal network coupled with

region pooling or can only detect and classify single actor per video, creating a challenge to adapt it to denser real-life scenarios.

2.2 Actor-action detection in videos

Actor-action detection problem is related to identifying the actors and their corresponding actions in a given clip, where both semantic localization and classification is required. The authors in [1] proposed the A2D dataset, a large scale benchmark dataset to study actor-action detection problem. An early approach of adaptive grouping of segments during inference improves segmentation in A2D [46]. [47] proposed weakly supervised method and train the model using only video-level tags. A two-stage model was proposed by [48], where objects are detected first and their bounding box are refined for segmentation outputs. [49, 50] use sentence priors to detect actor-actions in videos. Authors in [22] propose a joint end-to-end model which uses two-stream input (RGB + optical flow) to classify object regions and perform segmentation on them. Conceptually based on [51], they generate semantic features and use RPN to segment and classify actor-action pairs. [21] use similar approach and propose segmentation based region proposal and pooling to detect actor and action classes. They apply a region pooling based fully convolutional segmentation network for their actor segmentation, followed by 2D ResNet-101 [52] for action classification. Although prior works show great improvements on joint actor-action classification, they are limited by expensive region proposal and pooling which increases the approach's complexity.

Recent methods for action detection utilize pseudo-labeling [53, 10], multi-instance learning [54, 55, 53] and consistency regularization via data augmentation [56] to train with limited annotations. However, they often rely on external off-the-shelf object detector [30, 57, 10] or assume availability of a subset of fully annotated data to initialize training [56].

2.3 Weakly/semi-supervised learning

Weakly [58, 59, 54] and semi-supervised [56, 60, 61, 62] methods leverage limited annotations via consistency regularization [63, 62, 64] and pseudo-labeling [65, 66, 67] to train the model at the expense of reduced performance. Recent works on weakly and semi-supervised approach have shown comparable results in various tasks [61, 68, 69, 70, 71, 72] while reducing annotation cost drastically. The annotation cost to performance trade-off is justified as these methods reduce human effort and cost for large scale data annotation for various tasks (classification, localization, detection) [62, 65, 63, 56, 54, 73].

2.4 Limited label learning for action detection

Dense frame-wise spatio-temporal annotation is costly to obtain, therefore a natural step ahead was to use reduced annotations to train models for action detection task. Recent methods for action detection utilize pseudo-labeling [53, 10], multi-instance learning [54, 55, 53] and consistency regularization via data augmentation [56] to train with limited annotations. However, they often rely on external off-the-shelf object detector [30, 57, 10] or assume availability of a subset of fully annotated data to initialize training [56]. These works only use video-level annotation [53, 74, 54, 75] or point-label or pseudo-label [55, 76, 77] but they do not have any criteria for selecting the limited samples and can spend annotation budget selecting redundant and non-informative samples. A common drawback of these methods is the inferior performance compared to fully supervised methods, which limits their practical utility.

2.5 Active learning

Active learning (AL) has been used to iteratively select unlabeled data for assigning labels based on certain utility factors [78, 79, 80, 81, 82]. Labeling a large set of data can often prove to be expensive and unnecessary, which is why AL can be vital in selecting related unlabeled data for further annotation in an iterative fashion. AL algorithms use uncertainty [81, 83, 84, 85, 86, 87], entropy [8, 88], heuristics and mutual information [80, 89, 90], core-set selection [91, 92] to select samples which are most likely to provide maximum utility to the learning algorithm. AL based classification algorithms are effective for different modalities such as images [80, 81, 79], videos [93, 94, 95], text [96, 97, 98] and speech [99]. Classification only needs class labels for an entire sample, making the scoring easier for the algorithm. However, extending that to a complex task such as object detection is challenging as it requires dense annotations in each sample [8, 100, 101, 102]. Extending that to videos adds extra level of complexities as it requires spatio-temporal annotations and selecting parts of video for extra annotation via AL algorithm is challenging. [103] performs frame selection using AL for object segmentation but does not leverage temporal aspect of video for avoiding sequential annotation, increasing overall annotation cost. There are *no existing methods* which focus on the problem of *active sparse labeling* in videos for *spatio-temporal detection task* and existing deep AL approaches are not applicable directly for this task.

2.6 Omni-supervised learning

There has been some work on combining different type of annotations (box, points, tags, scribbles) via omni-supervised learning [104, 105, 106] to further reduce overall annotation cost. However, this is mostly focused on image domain. Annotating points and scribbles are less costly than boxes or masks while being usable as pseudo-labels together with box/masks [55, 104, 107]. Prior

works on image object detection [108, 109, 110, 104, 105] handle mixed label types using iterative [110, 109] and unified [104, 105] methods and require additional bounding boxes using off-the-shelf detector [33, 30] or pre-computed region proposals [104] for training.

For video action detection, [76] proposed using different labels (boxes, temporal point, tag) with the help of off-the-shelf detector [57], tracker [111] and linkers [112, 59]. They use only one type of label at a time and assume entire dataset has same label type for training. In contrast, we propose to use mixed labels (box, mask, scribble, tags, pseudo-labels) from partial data with an end-to-end action detection model that doesn't rely on any external detector or region proposal. Instead, we utilize a pseudo-label approach using superpixels [113, 114] to train a unified model that can handle mixed type of labels for training action detector.

CHAPTER 3: EFFICIENT ACTOR-ACTION DETECTION IN VIDEOS

The work in this chapter has been published in the following paper:

Aayush J. Rana and Yogesh S. Rawat. "We don't need thousand proposals: Single shot actoraction detection in videos." In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), January 2021.[115]

Actor-action detection in videos is a challenging problem where the goal is to detect all the actors in the video and determine which different actions are being performed by each actor. In this chapter, we explore a simple and efficient deep learning network to detect actors and their corresponding actions in videos. This network makes detection easier and faster using a streamlined single stage approach. We propose SSA2D, an encoder-decoder based unified network, which utilizes spatio-temporal contextual information between objects and their surrounding pixels for joint detection of multiple actors and corresponding activities in multiple input video frames at once.

We organize the chapter as follows: Section 3.1 describes the formulation of our efficient network and the objectives for training it, Section 3.2 provides the experiment setup, dataset details and the results for the proposed method, Section 3.3 gives analytical overview of different aspects of the method and Section 3.4 contains summary for this chapter.

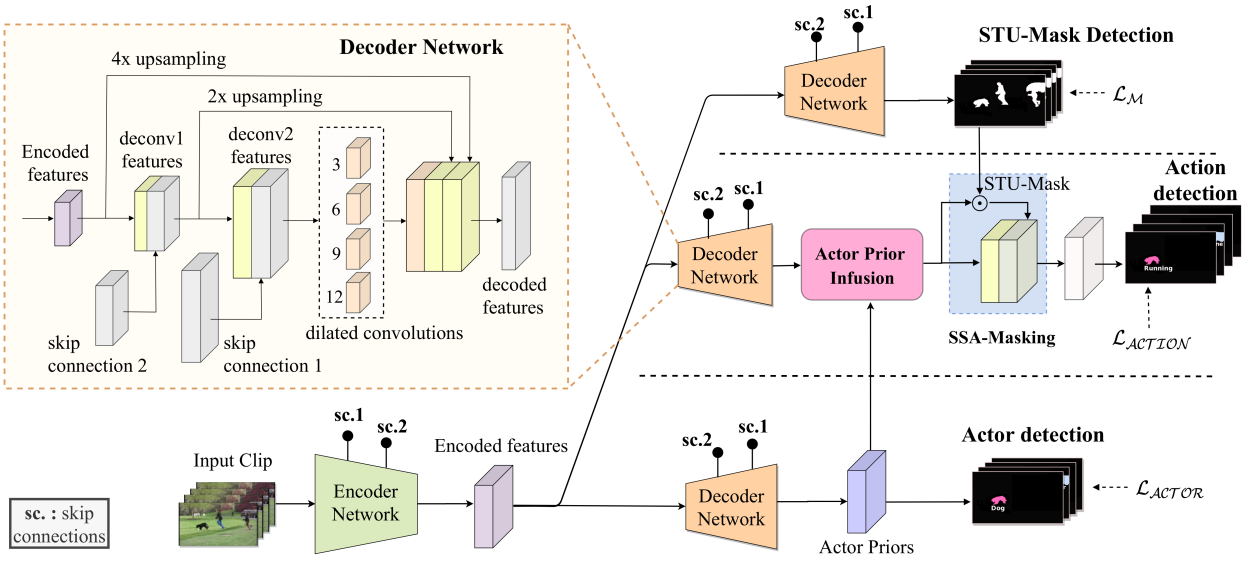

Figure 3.1: Overview of the proposed method for pixel-wise actor-action detection showing the overall SSA2D architecture. The 3D convolution based encoder network extracts features which are used in three separate branches for actor, action, and *STU-Mask* detection. The action detection branch uses actor prior infusion (*AP-Infusion*) to utilize actor-priors and *SSA-Masking* to focus on relevant activity regions in the video. All three branches use decoder network with similar architecture, however the weights are not shared among these branches.

3.1 Methodology

3.1.1 Problem formulation

Given a video $V \in \mathbb{R}^{T \times H \times W \times 3}$ as input, our proposed Single Shot Actor-Action Detection (SSA2D) network jointly predicts actor detection $ActorD \in \mathbb{R}^{T \times H \times W \times C_{actor}}$ and action detection $ActionD \in \mathbb{R}^{T \times H \times W \times C_{action}}$. Here, T is the number of frames in the input clip, H is the height and W is the width of the video frames, C_{actor} is total number of actor classes, and C_{action} is the total number of action categories. In addition to these two, SSA2D also predicts a spatio-temporal mask $STU - Mask \in \mathbb{R}^{T \times H \times W \times 2}$ that is used for filtering features relevant to objects of interest, re-

ducing computation of unrelated regions in dense video scenes. SSA2D consists of an encoder network E, and three separate branches for actor, action, and STU-Mask detection. Each task utilize a decoder network (D), which has similar architecture for all the three branches. An overview of SSA2D is shown in Figure 3.1.

3.1.2 Encoder network

Understanding and extracting relevant features both spatially and temporally is crucial in learning a video's actor-action relations. We utilize a *3D convolution* based encoder E that extracts actor and action related feature volume f_{enc} from a given input video clip $V \in \mathbb{R}^{T \times H \times W \times 3}$. The network takes as input a video clip of $T \times H \times W \times 3$ dimension with T frames at a resolution of H height and W width with 3 channels, and outputs a feature volume $f_{enc} \in \mathbb{R}^{\frac{T}{4} \times \frac{H}{16} \times \frac{W}{16}}$ as video encodings which has $\frac{1}{4}^{th}$ frames of the input with $\frac{1}{16}^{th}$ reduced spatial resolution. We use I3D [6] model as our encoder where we adapt the network by controlling the pooling strides (more details in supplementary). This encoder can use any state-of-the art 3D convolution network.

3.1.3 Decoder network

The spatio-temporal features f_{enc} extracted using the encoder network E needs to be decoded into a larger fine-grained resolution for jointly detecting the actors and actions, as shown in Figure 3.1. The decoder network D takes f_{enc} as input and performs a series of 3D deconvolution [116] and upsampling operations to get the desired resolution for fine-grained pixel-wise detection. We upsample the encoded features to $[\frac{T}{2} \times \frac{H}{4} \times \frac{W}{4}]$ as a final resolution to reduce parameters. We add skip connections from the encoder network to every deconvolution layer to preserve the suppressed features during downsampling. Adapting multi-scale object feature learning techniques from images, we extend atrous convolutions [117, 118] and feature pyramid network [119, 120] to

3D architecture for videos. Atrous convolutions helps encode multi-scale contextual information around each pixel while feature pyramid helps preserve features of smaller objects.

The same decoder architecture D is used in all the three branches, however, the network parameters are not shared as these branches solve different tasks. The final output from all the branches is up-sampled to match the resolution of the input video with the help of linear interpolation. A detailed architecture of D is shown in Figure 3.1 and more details are provided in the supplementary.

3.1.4 Actor detection

For pixel-wise actor detection, the actor detection branch utilizes encoded video features f_{enc} and learns pixel-wise actor prior (*A-prior* $\in \mathbb{R}^{T \times H \times W \times C_{ap}}$) for C_{ap} classes with the help of a decoder network D_{actor}. A final 3D convolution layer takes the learned *A-Prior* and predicts C_{Actor} channels for each pixel (C_{Actor} being the total number of actors present in the dataset). A *Softmax* activation is applied across actor channels for each pixel location as each pixel will correspond to only one of the actors. This gives us $ActorD$ for pixel-wise actor detection in the input video. The scores in each channel corresponds to one of the actor class and indicates its presence in that spatio-temporal location.

3.1.5 Action detection

The action detection branch $Action$ takes the spatio-temporal features f_{enc} from the encoder network E as input and uses the decoder architecture D_{action} from section 3.1.3 to learn action relevant feature maps f_a for action detection. As each actor's interaction with surrounding objects is decisive in inferring its actions, the actor detection branch will have more meaningful features corresponding to each actor. However, it is also important to focus only on the spatio-temporal

region where the action is occurring. To address these issues, we propose Actor Prior Infusion (*AP-Infusion*) and Single-Shot Attentive Masking (SSA-Masking), which allow the network to filter and learn meaningful interaction between the detected actors for action detection.

Actor Prior Infusion (*AP-Infusion*) The Actor Prior Infusion (*AP-Infusion*) provides additional information to the action detection network in form of latent actor representations. This is done by integrating *A-priors* with action related features, adding more actor focused contextual information and helps in action detection. As shown in Figure 3.1, the *A-priors* f_{ap} are integrated with action features f_a from the decoder network in $Action$ branch as $f_{act} = Conv3D(< f_a, f_{ap} >)$, where $Conv3D$ is 3D convolution operation and $<>$ represented feature concatenation. We also experimented with feature addition and observed similar performance.

Single-Shot Attentive Masking (*SSA-Masking*) Instead of generating proposal boxes from external networks [30] or using all possible region boxes [32], we use single-shot attentive masking for feature filtering. A fine-grained spatio-temporal region is helpful to filter and improve the coarse actor-action detection results. To get this spatio-temporal mask, the features f_{enc} from the encoder network E are passed to a decoder network $D_{STU-Mask}$ which predicts pixel-wise scores $STU - Mask \in \mathbb{R}^{T\times H\times W\times 2}$ for each spatio-temporal location in the input video. Each pixel's score in the *STU-Mask* indicates whether it is relevant to the action or not. The network learns to identify potential actor regions through the *STU-Mask*. This mask from the $D_{STU-Mask}$ is used as spatio-temporal unified mask $f_{mask} \in \mathbb{R}^{T\times H\times W\times 1}$ to filter the spatio-temporal features for action detection.

The action features f_{act} augmented with actor-priors are filtered using *SSA-Masking*. The augmented features f_{act} are integrated with the *STU-Mask* [$f'_{act} = f_{act} \odot f_{mask}$] to get the filtered features f'_{act}. The filtered features f'_{act} are integrated back with the original action features f_{act} [$f''_{act} =< f'_{act}, f_{act} >$] to keep both action as well as contextual background features for an effec-

tive learning. With this masking, forward pass only learns detection of useful feature regions while backward pass has minimal gradient update for unrelated regions. Furthermore, the *SSA-Masking* can be done on the whole frame in a single-shot, removing the need for extracting multiple region proposal boxes and performing ROI/TOI pooling. The masking can be done within the network, making this an end-to-end architecture. During training, we use the ground truth *STU-Mask*. While testing, we extract the $D_{STU-Mask}$ detection results and pass that as the *STU-Mask* within the network. Finally, the output feature from *SSA-Masking* is used to predict $ActionD$ with C_{Action} channels using 3D convolution, where each channel corresponds to one action class. Since each pixel is evaluated individually, it can be formulated to have multi-labels and multi-class predictions.

3.1.6 Objective function

The proposed network is trained end-to-end with joint learning of three tasks: actor detection, action detection, and STU-Mask detection. Since we predict pixel-wise maps for each branch, we have to consider the large imbalance in active and non-active pixels, with fewer active pixels for sparse scenes. This imbalance is handled using ratio loss for the scene. In case of image segmentation, this can be computed as a ratio of foreground pixels to background using the Generalized Dice Loss [121]. We extended it to videos as Generalized 3D Dice Loss with the following formulation:

$$\mathcal{L}_{\mathcal{DL}} = 1 - \frac{2\sum_{c=1}^{C}\sum_{i=1}^{N} p_{ci} * \hat{p_{ci}}}{\sum_{c=1}^{C}(\sum_{i=1}^{N} p_{ci}^2 + \sum_{i=1}^{N} \hat{p_{ci}}^2 + \epsilon)} \tag{3.1}$$

where the dice coefficient score is computed per class C of given task, N is total number of pixels in segmentation map of a video clip, probability $p_{ci} \in (0, 1)$ is the ground-truth segmentation map, and $\hat{p_{ci}} \in (0, 1)$ is the network's predicted segmentation map probability.

The actor detection loss is defined as the negative log-likelihood of the ground truth class and is computed as categorical cross-entropy per pixel. For C_{actor} set of actor classes, the actor detection

head generates C_{actor} segmentation maps, where each pixel's ground truth actor class is x and predicted actor class is $\hat{x}$. The loss is calculated for each pixel across all classes and then averaged over all pixels, which gives us the following loss formulation:

$$\mathcal{L}_{\mathcal{ACTOR}} = [-\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C_{actor}}(x_{i,j})log(\hat{x}_{i,j})] + \mathcal{L}_{\mathcal{DL}}. \tag{3.2}$$

Action detection is also defined similarly to actor detection, with C_{action} segmentation maps generated. For each pixel's ground truth action class y an action class $\hat{y}$ is predicted, and the loss is:

$$\mathcal{L}_{\mathcal{ACTION}} = [-\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C_{action}}(y_{i,j})log(\hat{y}_{i,j})] + \mathcal{L}_{\mathcal{DL}}. \tag{3.3}$$

We look at the STU-Mask detection task as a binary segmentation task, where all the actor pixels are considered as positive and all others as negative. The loss is computed using binary cross-entropy in combination with the dice loss:

$$\mathcal{L}_{\mathcal{M}} = [-\frac{1}{N}\sum_{i=1}^{N}p_i log(\hat{p_i}) - (1-p_i)log(1-\hat{p_i})] + \mathcal{L}_{\mathcal{DL}}, \tag{3.4}$$

where $\hat{p_i}$ is the prediction and p_i is the ground-truth. The total loss is a combination of these losses and is defined as:

$$\mathcal{L} = \mathcal{L}_{\mathcal{ACTOR}} + \mathcal{L}_{\mathcal{ACTION}} + \mathcal{L}_{\mathcal{M}}. \tag{3.5}$$

3.2 Experiments and results

3.2.1 Implementation and training details

We implement the proposed method in Keras [122] with Tensorflow backend. The encoder block uses I3D [6] pre-trained on Kinetics-400. We input a video clip of temporal resolution (T) of 16

frames and spatial resolution 224 x 224. The final output of the encoder network is 4 x 14 x 14, which we then upsample to 8 x 112 x 112 for *STU-Mask* detection branch and 8 x 56 x 56 for actor and action detection branch. For the RGB + optical flow approach, we perform two stream implementation where two encoders are used for each input type. The encoders share some of the final layers to reduce network size, and skip connections are passed from both streams. We further provide the encoder and decoder network details.

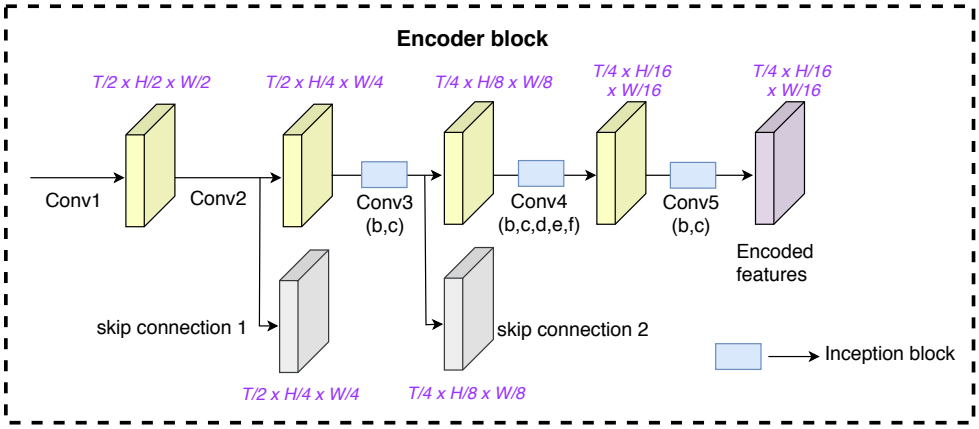

Figure 3.2: Encoder block details. The encoder contains multiple inception blocks to extract relevant features for decoder network. To retain fine grained features from initial layers, two skip connections are passed to the decoder network, which helps in fine grained pixel-level detection.

Encoder details As shown in Figure 3.2, the input clip is passed through the I3D based backbone network. From Figure 3.2, *Conv1* and *Conv2* are convolutional layers with same kernel size and strides as the I3D network. *Conv3*, *Conv4* and *Conv5* use the inception configuration with 2, 5 and 2 inception blocks respectively. We change the pooling strides and kernel sizes accordingly to get final output of $\frac{T}{4} \times \frac{H}{16} \times \frac{W}{16}$ from an input clip of $T \times H \times W$. We take skip connection after *Conv2* layer and *Conv3* layer, which is passed to each of the decoder block accordingly.

Decoder details The purpose of decoder block is to take encoded features and produce detection masks accordingly. All three branches(*STU-Mask* detection, actor detection, action detection) of

the proposed network use identical decoder block. Only the final output layer's channel is adjusted according to the desired branch output. As A2D only provides annotation for single frame, loss for action detection is only computed for single frame. As such, decoder block in actor detection is configured to predict only single frame output. However, the object features going into the OFI block does not get affected by this configuration. Since ViDOR dataset has per frame annotation, the decoder block outputs all frames predictions. Input to the block is encoded features from the encoder block. This is passed through deconvolution layers which will perform convolution as well as upsampling of the layers to increase the size. The operation is costlier than only convolution, so we only implement two deconvolution layers. To help retain features, we add skip connection from the encoder block. *Skip connection 2* is concatenated with *Deconv1* features and *skip connection 1* is concatenated with *Deconv2* features. The output feature size of *Deconv2* layer is adjusted to be $\frac{T}{2} \times \frac{H}{4} \times \frac{W}{4}$, which is temporally half and spatially one-fourth of the input resolution. This was done to keep the network smaller and improve efficiency. We apply dilated (atrous) convolution to capture features at multiple receptive fields (rate=3,6,9,12). Following feature pyramid network, we also take features after *Deconv1* and *Deconv2* layer and upsample it *4*$\times$ and *2*$\times$ respectively, which is then concatenated to features from dilated convolution.

Optimization We use Adam optimizer [123] with an initial learning rate of 1e-4 and finetune at a rate of 1e-5. For our joint training task, we can fit an effective batch size of 14 clips per iteration. The model is trained for 5 epochs with initial learning rate and fine-tuned for another 6 epochs.

Joint training We train all three branches together with the loss weights assigned based on class distribution per task. For the A2D dataset, *STU-Mask* detection is given the weight of 0.3, while both actor and action detection task is given weights of 1.3 (based on per class pixel distribution).

STU-Mask We input the *STU-Mask* of size 4 x 56 x 56 for the action detection task, which helps to increase focus on the related pixels. For training, we use all actor regions from ground truth as

the *STU-Mask*. During inference, we use the *STU-Mask* predicted by the *STU-Mask* branch.

Since our network does not have any fully connected layers or an extra region proposal network, the network has fewer number of parameters and can be trained end-to-end in a single stage.

3.2.2 Datasets

A2D dataset A2D [1] is the first video dataset with multiple actor classes and action classes in the same clip along with semantic labels. It provides pixel-level semantic labels of 3-5 frames for each video and is the only joint actor-action segmentation benchmark reported in prior works [1, 46, 48, 22]. The dataset consists of 3,782 YouTube videos, consisting of 7 actor classes performing one of the 9 action classes. A total of 43 actor-action pairs are valid and used for joint actor-action segmentation task. Both pixel level and bounding box annotation per actor-action pair are provided in this dataset.

VidOR dataset We also evaluate our method on the VidOR dataset [124] which contains 10,000 videos with 80 object categories and 42 action predicates with bounding box annotations. Although it has more videos for training, the dataset is more challenging as it has a wide range of objects with a skewed distribution where 92% of objects are from only 30 categories. Each action is part of a triplet and consists of a subject and an object, with the subject performing the action. Thus, action detection using object and its surrounding context is more meaningful.

3.2.3 Evaluation metric

Following the evaluation protocols from [46] and [22], we measure average per-class accuracy (ave), global pixel accuracy (glo) and mean pixel Intersection-over-Union $(mIoU)$ as evaluation metrics. Accuracy is the percent of pixels with correct label prediction, where (glo) is computed

over all pixels and (ave) is first computed per class and then averaged. Since background covers a large area and most models are biased towards background, mIoU is the most representative metric for correct pixel prediction over all classes [48]. We report results for actor, action and joint actor-action detection for (ave), (glo) and $(mIoU)$ metrics for a fair comparison with existing methods.

Table 3.1: Quantitative comparison of SSA2D on A2D dataset with prior approaches using RGB and RGB + optical flow (OF) as input, reporting average per-class accuracy (ave), global pixel accuracy (glo) and mean pixel Intersection-over-Union $(mIoU)$ for each task. We also report the processing time per frame in millisecond for each method. † Uses sentence priors. **Uses weakly-supervised training.* ** Is time adjusted for same hardware setting by correspondence with authors.

Input	Method	Actor			Action			Joint (A,A)			Time
		glo	ave	mIoU	glo	ave	mIoU	glo	ave	mIoU	(ms)
RGB	GPM + TSP [46]	85.2	58.3	33.4	85.3	60.5	32.0	84.2	43.3	19.9	-
	GPM + GBH [46]	84.9	61.2	33.3	84.8	59.4	31.9	83.8	43.9	19.9	-
	Chen et al. [125]*	91.3	49.2	49.2	87.4	35.1	38.7	87.1	43.1	26.7	-
	Ji et al. [22]	93.7	79.5	66.5	86.3	60.4	36.8	87.8	46.2	29.4	-
	Dang et al. [21]	95.0	85.5	67.0	92.9	68.8	48.1	92.5	51.5	34.5	750
	SSA2D (Ours)	96.1	79.4	66.8	94.4	66.2	46.5	93.8	49.3	34.6	67
RGB + OF	TSMT + GBH [48]	85.8	72.9	42.7	84.6	61.4	35.5	83.9	48.0	24.9	-
	TSMT + SM [48]	90.6	73.7	49.5	89.3	60.5	42.2	88.7	47.5	29.7	-
	Gavrilyuk et al. [49]†	92.8	71.4	53.7	92.5	69.3	49.4	91.7	52.4	34.8	-
	Ji et al. [22]	94.5	79.1	66.4	92.6	62.9	46.3	92.5	51.4	36.9	350**
	Dang et al. [21]	95.3	86.0	68.1	93.4	70.7	51.1	93.0	56.4	38.6	1100
	SSA2D (Ours)	96.2	80.1	67.5	94.9	69.1	51.3	95.0	54.7	39.5	180

3.2.4 Results

The performance of SSA2D on A2D is shown in Table 3.1. Using only RGB stream, SSA2D gives improved joint actor-action mIoU with significant reduction in inference time *(∼11x faster)*. This demonstrates that the network's joint training is able to learn action features based on actors

while computing video level detection faster than previous methods. Moreover, using RGB+OF input we observe that the network gives improved mIoU scores on action and joint task as expected, demonstrating that our approach generalizes to different types on input modalities. We also analyze per class performance and the scores are shown for both of our RGB model and RGB+OF model in Figure 3.3. We observe that the proposed method can detect most classes accurately in RGB model and the scores are further increased with additional flow information.

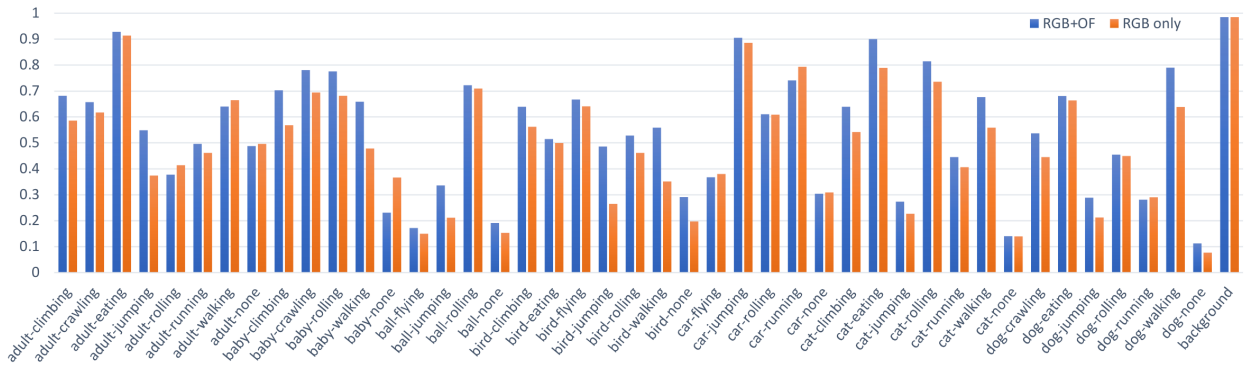

Figure 3.3: Per actor-action category average accuracy score for A2D. The orange bars show results for RGB modality and blue bar for combined RGB and Optical Flow. We observe that on average, most of the classes benefit from having extra optical flow information.

We report the performance of proposed method on VidOR dataset in Table 3.2. We evaluate the model on same evaluation metrics as used for the A2D dataset. Due to its long tail distribution, the dataset suffers from large data unbalance. As such, even when our network performs well on those classes, the average accuracy and mean IoU scores drop due to tail classes with fewer training samples.

Qualitative evaluation Figure 3.4, 3.5 and 3.6 show qualitative results for actor-action detection. We observe that the proposed method can predict reasonable detections for most of the cases. Figure 3.7 shows that the network predicts correctly even though ground truth annotation is missing labels. The network is able to generalize and learn effective actor-action features to predict the

Figure 3.4: Qualitative results of our method on A2D dataset. The first row shows the input key frame, second row shows the ground truth with annotation labels and third row shows our joint actor-action detection result with predicted labels.

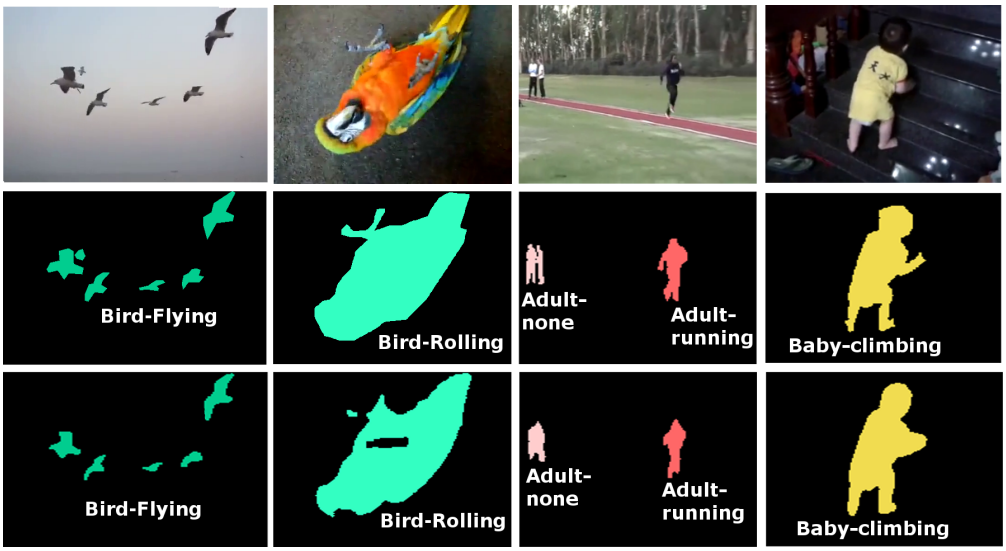

Figure 3.5: More qualitative results of our method on A2D dataset. The first row shows the input key frame, second row shows the ground truth with annotation labels and third row shows our joint actor-action detection result with predicted labels.

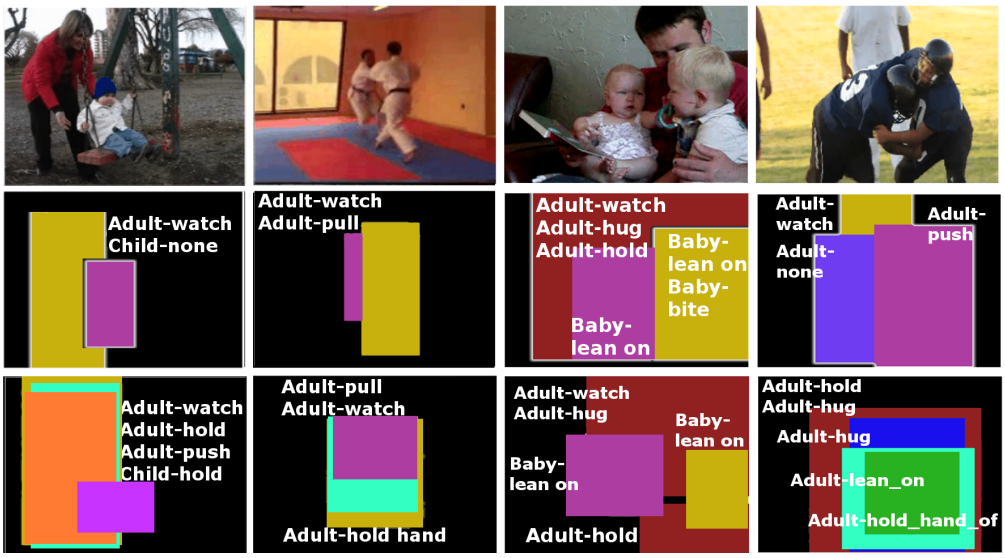

Figure 3.6: Qualitative results of our method on ViDOR dataset. The top, middle and bottom row represents input key frame, ground truth and our joint actor-action predictions with label respectively.

missing labels. The last column shows the network detecting a hard sample correctly. Even though the cat blends with the background, it is well segmented and detected as *cat-jumping* class. Using 3D convolution on videos where the object is better visible in other frames, detection improves in such challenging frames as features is evaluated together for the entire video.

3.2.5 Ablation Studies

We further validate the importance of different components proposed in our model through ablation experiments. Since our contribution is agnostic to input type, we evaluate all variations against the full RGB only model in Table 3.2.

Actor Prior Infusion (*AP-Infusion*) One of the key components in improving action detection in our model is the use of actor prior for inferring activities. The *A-prior* coming from actor detection

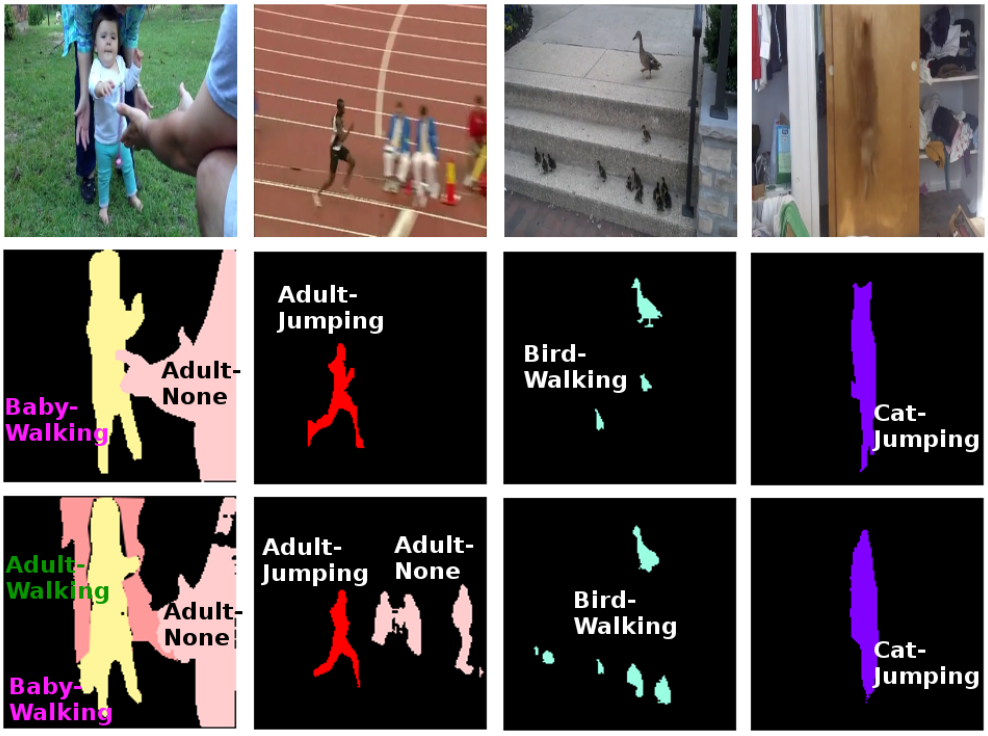

Figure 3.7: Qualitative analysis of some success cases where the network predicts better than the ground truth. The top, middle and bottom row represents input key frame, ground truth and our joint actor-action detection predictions with label respectively. The network correctly predicts labels for actions which are not annotated in the ground truth but present in the clip.

branch provides contextual information regarding all actors around each pixel. It is reasonable to have an understanding of the actors involved in order to better judge the actions happening. While [35] shows that using pair-wise actor features helps improve action classification in images, our *AP-Infusion* approach uses all of the involved actor's features together because of the pixel-level detection. We train our model without using the *AP-Infusion* to evaluate its effectiveness. As seen in *Full* and *w/o Actor-prior* models of Table 3.2, *A-prior* provides a significant gain in the action detection task ($\sim 6\%$ improvement in mIoU for A2D) and subsequently increases the scores for all other tasks. Since we perform a joint training, we also observe the decrease of scores for actor detection task when feedback from the *AP-Infusion* block is not present.

Table 3.2: Ablation study of various components of SSA2D and their effect on actor-action detection on A2D and VidOR dataset. We report scores on average per-class accuracy (ave), global pixel accuracy (glo) and mean pixel Intersection-over-Union $(mIoU)$. Values in bracket represent scores for the 20 most frequent classes in VidOR dataset.

Dataset	Method	Actor			Action			Joint (A,A)		
		glo	ave	mIoU	glo	ave	mIoU	glo	ave	mIoU
A2D	**Full (RGB only)**	**96.2**	**80.1**	**67.5**	94.4	**66.2**	**46.5**	**93.8**	**49.3**	**34.6**
A2D	w/o *Actor-Prior*	96.1	79.1	65.7	93.9	61.5	40.9	93.4	46.3	32.1
A2D	w/o *SSA-Masking*	96.1	79.9	67.2	**94.6**	63.9	43.6	93.7	48.2	33.8
A2D	w/o atrous convolutions	92.4	76.4	62	94.1	62.3	41.6	92.8	45.2	31.7
A2D	w/o multi-scale	96.0	79.8	66.5	94.2	63.8	43.1	93.6	47.9	33.1
VidOR	**Full (RGB only)**	**72.2**	**7.6**	**5.1**	**66.8**	**33.2**	**7.9**	**41.7**	**15.7**	**2.1**
		(54.1)	(20.5)	(12.5)	(70.2)	(40.8)	(11.7)	(44.2)	(18.8)	(5.1)
VidOR	w/o *Actor-Prior*	71.1	7.1	4.1	61.8	28.3	5.9	37.7	12.1	1.1
VidOR	w/o *SSA-Masking*	71.8	7.1	4.3	65.4	31.7	7.1	39.2	15.2	1.8
VidOR	w/o atrous convolutions	71.5	6.4	3.4	63.1	30.8	6.2	38.3	14.4	1.3
VidOR	w/o multi-scale	71.2	6.1	3.1	61.7	29.5	6.0	37.8	14.1	1.1

SSA-Masking *SSA-Masking* is used in action detection task to filter and enhance focus on action regions for pixel-wise detection. This reduces the surplus background noise and helps in a faster convergence. Our motivation to use the *STU-Mask* is to provide emphasis on features related to actors while filtering out excess background data. In RPN based methods, ROI-Pooling play the role of feature filtering. However, pooling is performed for each proposal independently making it computationally expensive. We use a unified mask for all the actors in the scene for this filtering making SSA2D more efficient. SSA-Masking enables the network to focus more on the actor pixels while suppressing the background pixels, which leads to an improved network performance for action detection ($\sim$ 3% increase in mIoU for A2D) and also provides a faster network convergence ($\sim$3x).

3.3 Analysis

Comparative analysis Figure 3.8 shows a comparative view of our method along with [22, 46, 21] in terms of performance and speed. Compared to [21], our training does not use weights pre-trained on segmentation task and trains the decoders from scratch, while [21] uses pre-trained weights on segmentation tasks. We observe that our method performs significantly better compared to [22, 46] in all evaluation metrics as seen in Table 3.1. We see that despite fast inference time for [22], it under-performs and has a larger model. Furthermore, our quantitative scores are similar or slightly better than previous state of the art method [21] and has significantly better inference time($\sim$11x). This large gap in inference time makes our approach better suited for actor-action detection in videos as compared to all prior works.

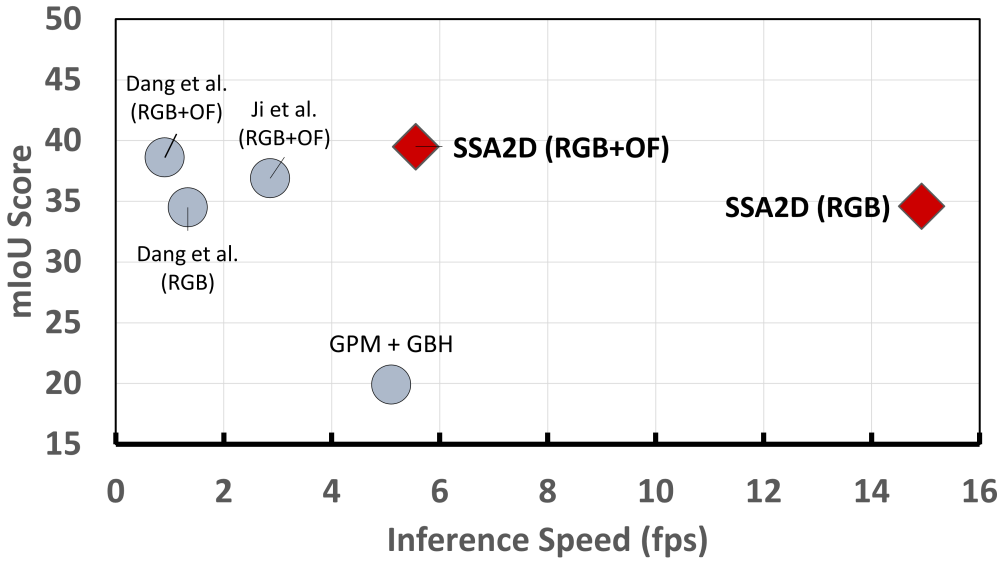

Figure 3.8: A comparative analysis of SSA2D with existing methods in terms of speed and performance. We observe that SSA2D is faster with comparable performance. The x-axis represents inference speed in frames per second and the y-axis represents the mean pixel-wise intersection over union score for joint actor-action detection. (RGB+OF - Using both RGB and optical flow).

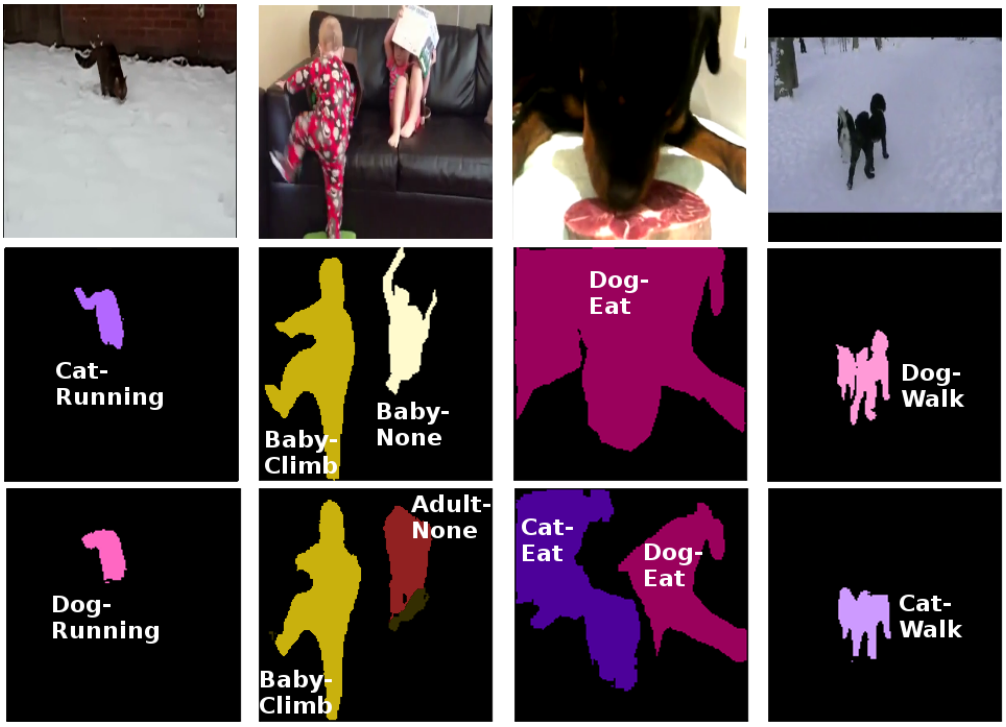

Figure 3.9: Qualitative analysis of some failure cases. The top, middle and bottom row represents input key frame, ground truth semantic segmentation mask and our joint actor-action detection predictions with label respectively.

Network parameters Another key aspect of the proposed method is the smaller network size (35M params for RGB and 55M params for RGB+OF) compared to [21, 22] (44M params for RGB and 88M params for RGB+OF). Compared to prior works, SSA2D has reduced network size which relates to the overall efficiency and performance speed. The memory-efficient reduced network also enables end-to-end training for all tasks simultaneously as compared to multi-stage training [21], which is time consuming.

Running time A crucial difference between SSA2D and prior works is that previous works rely on RPN as an auxiliary task during training to obtain actor regions for ROI pooling. Our method uses end-to-end pixel-wise detection and jointly trains actor-action tasks on pixel level while keeping

the implementation efficient and effective. For a fair evaluation, we evaluate the time taken to perform the evaluations on a single core of an Intel Xeon 2.3GHz CPU using a single NVidia Tesla K80 GPU [21]. During inference, our system takes 180 *ms per frame* with the RGB + OF model, while it takes only 67 *ms per frame* for single stream RGB model. [21] report computational time of 1100 *ms per frame* for their full system with optical flow, with around 350 *ms* being used for optical flow estimation.

Failure cases In Figure 3.9 we have shown some of the failure cases of our method. We observe that the network is able to detect correct foreground region in most cases, however, it gets confused on similar actors such as dog-cat or adult-baby. The approach suffers from data imbalance so classes with lower samples will perform lower which is observed in the prior works as well. This is one of the limitation of our approach which can be improved in future works.

3.4 Summary

In this chapter, we propose SSA2D, a simple yet effective approach for single-shot actor-action detection in videos. We demonstrate that actor-action detection in videos can be performed without relying on region proposal network where thousand of proposals are required making it in-efficient for dense video scenes. We evaluate the proposed approach on A2D and VidOR datasets and achieve comparable (sometimes even better) performance when compared with prior works. The proposed model can be efficiently trained (2x faster) with a fast inference ($\sim$11x faster for RGB and $\sim$6x faster for RGB+optical-flow) with fewer network parameters when compared with best performing prior works.

CHAPTER 4: VIDEO ACTION DETECTION USING SPARSE LABELS

The work in this chapter has been published in the following paper:

Aayush J. Rana and Yogesh S. Rawat. "Are all frames equal? Active sparse labeling for video action detection." In Proceedings of the 36th International Conference on Neural Information Processing Systems (NeurIPS), 2022.[25]

In the previous chapter, we demonstrated an efficient end-to-end network that can predict actor-action detection in videos with faster inference time. In this chapter, we focus on improving video action detection efficiency by reducing annotation cost of the dataset. This is a challenging problem as video action detection models require lots of annotated frames to train properly, but annotating such large dataset is costly. We study ways to reduce annotation cost of large video datasets by labeling frames sparsely for the task of efficient video action detection. We explore different ways to select a few frames for annotation that improves video action detection while keeping annotation cost low. We also propose methods to better train models using such sparsely annotated dataset.

We organize the chapter as follows: Section 4.1 describes the proposed methodology, Section 4.2 demonstrates the experiment setup and results, Section 4.3 gives the analysis of components in the proposed method, and Section 4.4 concludes the chapter with a summary.

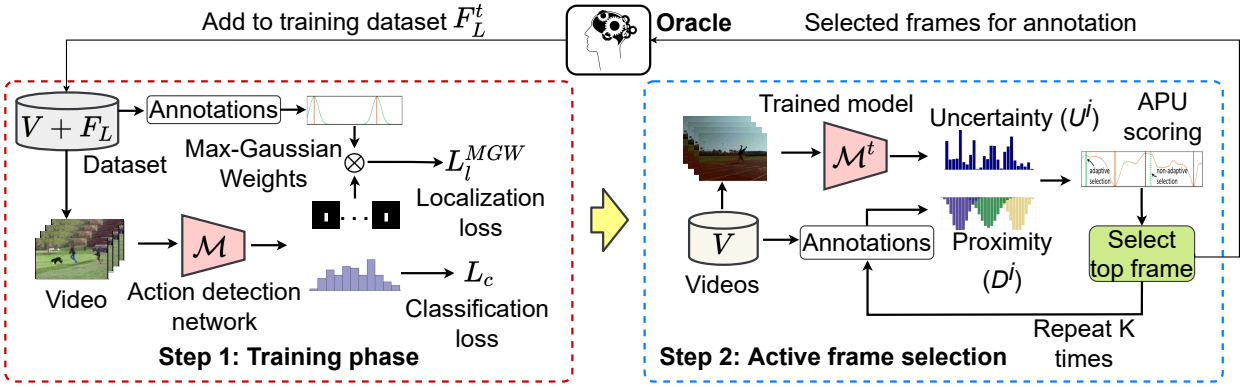

Figure 4.1: Overview of proposed approach. It consists of training and selection. During training the network is trained using existing annotations from the training set using *MGW-loss* to handle the sparse annotations. During iterative *APU* selection phase, the trained network is used to predict localizations on each frame of videos in the training set. Using these predictions, *APU* computes a score for each frame in a video to rank them and top K frames are sent to oracle for annotation.

4.1 Methodology

4.1.1 Problem formulation

We aim at reducing the annotation cost for labeling a set of videos $\mathcal{V} = \{V_1, ...V_N\}$ with N videos to learn an action detection model $\mathcal{M}$, as shown in Figure 4.1. We start with an initial set of sparse labels $\mathcal{S}_L^0 = \{V_{cls}, F_L^0\}$ that consists of annotated frames with class label V_{cls}, where only a small number of frames F_L^0 are annotated. This initial set of sparsely annotated videos is used to initialize an action detection model $\mathcal{M}^0$. This initialized model $\mathcal{M}^0$ is then used to estimate a utility score for all the unlabeled frames F_U^0 from the set of videos $\mathcal{V}$. The goal is to automatically select frames from unlabelled set to be manually labeled and obtain new set of sparse labels $\mathcal{S}_S^0$ which is merged with $\mathcal{S}_L^0$ for a new labeled set $\mathcal{S}_L^1$. The number of additional frames are selected based on a total budget B and they are annotated by an oracle (e.g. human annotator). The action detection model $\mathcal{M}$ is retrained using the new annotation set $\mathcal{S}_L^1$ and an updated model $\mathcal{M}^1$ is obtained. This

process is repeated until we find a set $\mathcal{S}_L^F$ with several annotated frames in the videos $\mathcal{V}$ such that $\mathcal{M}^{\mathcal{F}}$ meets the target performance or the total budget B is exhausted.

4.1.2 Active sparse labeling

4.1.2.1 Sparse labeling

We hypothesize that some frames will have more utility than others for learning action detection due to several factors, such as lack of motion, variation in action dynamics, redundancy in appearance or redundancy in action. In sparse labeling, we annotate only l frames $f_{v,l}$ in a video v instead of labelling all of them, leaving a set of u unannotated frames $f_{v,u}$. Therefore, it avoids annotation of frames with lower utility and helps in reducing the overall labeling cost. Each video v has a class label, denoted as v_{cls}, for the action category and a set of l annotated frames $f_{v,l}$ which indicates the localization of actions.

4.1.2.2 Uncertainty as frame utility

In each AL cycle, our goal is to select video frames for labeling which will have the highest utility for learning action detection. Uncertainty provides a measure to estimate models confidence on its decision and has been used for selecting informative samples in existing works [126, 127, 128, 95]. These works are focused on classification and therefore the uncertainty is computed for the entire sample. Instead, we require informativeness of each frame in a video which is different from these works as it is computed for partial sample. The action detection model $\mathcal{M}$ provides spatio-temporal localization for the entire video and we propose to use the pixel-wise confidence score of localization on each frame to estimate frame-level uncertainty. We use MC-dropout [7] to estimate the model's uncertainty for each pixel in the video. MC-dropout is a more efficient form

of uncertainty estimation compared to using a Bayesian neural network and is easier to implement [7, 67]. The uncertainty is estimated over T different trials and this score is averaged over all the pixels in a frame. For a given video v with I frames, the uncertainty U^i for the i^{th} frame over T trials is computed as,

$$U^{i\in[1,I]} = \frac{1}{I^p}\sum_{h=1}^{I^p}\frac{1}{T}\sum_{j=1}^{T} -log(P(v_h^i, j)) \tag{4.1}$$

where I^p is the total number of pixels in a frame, and $P(v_h^i, j)$ represents the model prediction for h^{th} pixel in the i^{th} frame of video v during the j^{th} trial.

4.1.2.3 Adaptive proximity-aware uncertainty

Unlike images, the motion in videos has some continuity and it is highly likely that the frames close to each other will have similar uncertainty scores. Therefore selecting frames merely based on uncertainty will favor adjacent frames which may have similar utility for learning action detection. To overcome this issue, we propose a selection mechanism, termed as *Adaptive Proximity-aware Uncertainty (APU)*, which ensures that the selected frames have diversity in the temporal domain. APU scoring incorporates a distance measure into cost estimation and uses their *proximity* to the existing annotated frames. As we select more frames, this distance measure should *adapt* to the additional selected frames. We use a normal distribution $\mathcal{N}(\mu,\, \sigma^2)$ for distance measure D, where each annotated frame has its own distribution centered around its temporal location in the video. Given a video with K annotated frames, the distance measure D^i for the i^{th} frame of the video is computed as,

$$D^i = 1 - \sum_{j=1}^{K} \varphi_i^j e^{-\frac{1}{2}(\frac{i-\mu_j}{\sigma})^2}. \tag{4.2}$$

where D^i is distance measure for unannotated frame i from annotated frame j, the distribution

$\mathcal{N}$ for j^{th} annotated frame is centered at frame j with μ_j mean and σ variance, and $\varphi_i^j \in [0,1]$ is the mask to select the closest distribution for i^{th} frame. The value of mask φ_i^j will be 1 for j^{th} distribution if it is closest to the i^{th} frame, otherwise it will be 0. APU scoring uses both uncertainty and proximity and therefore prefers frames with high uncertainty and ensures temporal diversity. The overall APU score $\mathcal{U}_{APU}^i$ for a given frame is computed as,

$$\mathcal{U}_{APU}^i = \lambda U^i + (1-\lambda) D^i \tag{4.3}$$

where λ is used to control the contribution from uncertainty and temporal diversity. It is set to 0.5 for equal contribution in our formulation where U, D are both normalized in range (0,1).

4.1.2.4 Informative frame selection

Once we get $\mathcal{U}_{APU}$ score for all the frames in $\mathcal{V}$ videos, we select the frame with highest score globally and then score the remaining frames again with the adapted distance measure. The re-scoring is necessary to reduce probability of picking frames around same region, since a region doing poorly is likely to have more frames which scored higher in the selection process. We only need to recompute the distance measure, which is computationally inexpensive. The entire selection algorithm is provided Appendix. Once we have F_{annot} frames selected as per our budget, they are annotated by an oracle and the training set is updated with these new annotations. This completes one AL cycle and the model $\mathcal{M}$ is trained using the updated annotations.

4.1.2.5 Non-activity suppression

If all pixels in a frame are considered to compute its utility, non-activity regions may negatively influence the score as the model easily determines background pixels compared to the actual action region in a frame. A low uncertainty score from background pixels will lower the overall frame

uncertainty even if the activity region has high uncertainty, specially in videos with a relatively larger background area as compare to the actual action region. Therefore, we ignore pixels which are predicted as background (true negatives and false negatives) with high confidence (using a threshold τ) when computing the frame-level uncertainty. This might exclude some foreground pixels (false negatives) from the uncertainty estimation. However, these pixels will not be useful even if we use them as they are predicted as background due to low uncertainty.

4.1.3 Objective function

Learning from sparse labels Given a video clip $V = \{f_1, f_2, ...f_N\}$ with N frames where K frames are annotated such that $K < N$, we have to detect the action through the entire clip. A traditional action detection network is trained with the help of two different objectives, a classification loss L_c for action category and a localization loss L_l for spatio-temporal detection [23]. The classification loss L_c is computed for the entire video clip and the localization loss is computed for every frame in the video.

Sparse labeling will not allow us to compute the localization loss L_l on every frame due to missing annotations. The localization loss L_l with sparse labeling can be computed as, $L_l = \sum_{i=1}^{N} \beta^i L_l^i$. Here, L_l^i represents localization loss in the i^{th} frame and $\beta^i \in [0, 1]$ indicates masking, which will be 1 for annotated frames and 0 otherwise. The masking strategy only uses the annotated frames for learning, therefore it is not quite effective. In a contrastive approach, we can use all the frames for learning by generating their *pseudo-labels* with the help of interpolation of annotations from neighboring frames. This will allow us to use all the frames but incurs noise from the pseudo-labels.

Max-gaussian weighted loss We propose a simple loss formulation which benefits from both, masking and pseudo-labels. We hypothesize that the pseudo-labels close to ground-truth labels

will be more reliable. Based on this, we propose *Max-Gaussian Weighted Loss (MGW-Loss)* which discounts the approximated pseudo-labels as they will not be as reliable as the actual ground-truth. We compute localization loss for each frame using both available and pseudo-labels, where the pseudo-labels have a varying weight in the overall loss component. The approximated annotations will not have a similar weight as their distance from the annotated frames will vary. We use a mixture of Gaussian distribution $w \in \{1..W\} \sim \mathcal{N}_w(\mu_{gt}, \sigma^2)$ to assign the weight to each frame, given $gt \in \{1..K\}$ actual ground-truth frame location as the mean of the distribution and σ is the variance of the distribution. We define the weighted localization loss L_l^{MGW} as,

$$L_l^{MGW} = \sum_{i=1}^{N}(\sum_{j=1}^{K} \phi_j^i e^{-\frac{1}{2}(\frac{i-\mu_j}{\sigma})^2}) L_l^i. \tag{4.4}$$

Here L_l^i is the localization loss of i^{th} frame for any video, μ_j is the frame location for j^{th} annotated frame, and $\phi_j^i \in [0, 1]$ is the mask to select the max distribution for i^{th} frame. The value of mask ϕ_j^i will be 1 for j^{th} distribution if it has the maximum probability among all Gaussians at location of i^{th} frame, otherwise it will be 0. The value of σ controls the weighting mechanism and it has two extremes. The high variance is equivalent to interpolation where all the frames will have equal weights and low variance is equivalent to masking where weights of pseudo-labels will be 0.

4.1.4 Action detection model

Video action detection is a challenging problem and the existing methods usually follow a complex pipeline [23, 24, 20]. Region proposal based approach has been found to be exceedingly effective [44], which has also been extended to tube proposals [23, 24]. However, training these two-step methods is not efficient, specially when we have to develop an iterative framework for AL. We follow a simpler approach where classification and detection can be done in an end-to-end training [5]. We simplified VideoCapsuleNet [5] further and replaced the 3D routing with 2D routing [129] which makes it more efficient in terms of memory and training speed. We then added dropout

layers for uncertainty and used *MGW-Loss* from Eq. 4.4 to handle sparse labels. To handle sparse labels, we get the frame-wise weight from Max-Gaussian Weighted method and adjust the loss using this weight. Following [5], the network is trained using spread loss for classification and binary-cross entropy loss for spatio-temporal localization.

4.2 Experiments and Results

4.2.1 Implementation details

We implement our method in PyTorch [130]. In video action detection architecture, we use I3D encoder head [6] with pre-trained weights from the Charades and Kinetics dataset [131]. We use Adam optimizer [123] with a batch size of 8 and train for 22K iterations in each active learning cycle. We train our model using a single *16GB Nvidia RTX 5000 GPU*. The frame selection method only runs in inference mode with *Dropout* enabled, thus using only a fraction of the GPU memory. Due to this, we can run multiple instances in parallel for frame selection in the training video set, reducing the time taken for frame selection process. During each iteration, we only select the given percentage of frames for further annotation and we retain the previous set of annotated frames. On a 8 core 3.2 GhZ Intel CPU and 16GB Nvidia RTX 5000 GPU combination, each AL frame selection round takes **50 minutes** for UCF-101. The model training for UCF-101 takes about **15 minutes** per epoch, which is trained for 40 epochs for each set of annotations.

For YouTube-VOS task, we use two existing methods [4, 9]. We use $\tau = 0.9$ for non-active suppression and $\sigma = 1.3$ for Eq. 4.2 and Eq. 4.4, which were empirically determined.

Action detection network We use the 2D variant of video capsule network [5] for action detection task on UCF-101-24 and JHMDB-21 dataset. The network takes an input clip of $T \times H \times W \times C$ dimension [T=frames, H=height, W=width, C=channels] and outputs T frames of $H \times W \times 1$

dimension. It also predicts the class prediction vector for the entire clip. The 2D capsule network takes a batch size of 8 samples per iteration, with each sample clip of size $8 \times 224 \times 224 \times 3$ with a temporal skip rate of 2. We follow the same input/output format as the original paper for 3D capsule network [5] for the 2D network variant. The full architecture detail is shown in figure 4.2.

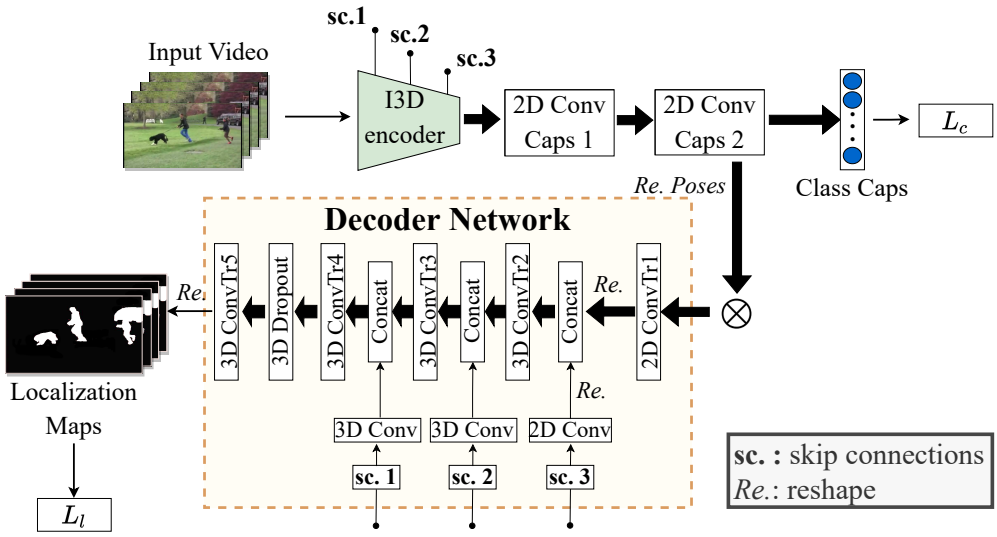

Figure 4.2: Overview of the proposed action detection network. Based on [5], features are extracted from input frames using I3D [6] architecture based encoder. We take features from $Mixed\_4f$ layer of I3D network. This is then followed by two 2D convolutional capsule layers which outputs class capsules. The class capsules is used for final class prediction and classification loss computation. This is followed by series of transpose convolution layers (2D and 3D) for upscaling the feature map and concatenation with features from intermediate layers of the I3D encoder via skip connections. We finally obtain the localization maps of same size as input video, which is used for detection loss.

4.2.2 Sparse learning settings

In the initialization stage, we assume the availability of annotations for $I\%$ of frames in each video in $\mathcal{V}$ to make sparse annotation set $\mathcal{S}_L^0$. These frames are randomly selected for the first stage. We use 1%, 3%, and 5% initial frames for UCF-101, J-HMDB, and Youtube-VOS respectively,

determined empirically based on each dataset size. We assign annotation cost for each frame as $C_{frame} = Actor \times Clicks$ based on clicks per actor (bounding box/pixels).

Interpolation: The annotation interpolation for UCF-101 is done using linear interpolation of the bounding box corners. The pixel-wise annotation interpolation for J-HMDB is done using CyclicGen [132]. In case of edge frames or single frame annotations, we extrapolate the annotation to other frames.

4.2.3 Datasets

We evaluate our approach on three different datasets, **UCF-101** [3], **J-HMDB** [2] and **YouTube-VOS** [4]. UCF-101 has 3207 untrimmed videos from 24 different classes with spatio-temporal bounding box annotations. J-HMDB dataset contains 928 trimmed videos from 21 classes with spatio-temporal pixel-level mask annotations. We further evaluate our method on **YouTube-VOS** [4] for video object segmentation to demonstrate its generalization capability. YouTube-VOS consists of 3471 training videos (65 categories) with pixel-level annotation for multi-object segmentation.

4.2.4 Evaluation metrics

Following prior action detection works [5, 133, 77] on UCF-101 and J-HMDB datasets, we compute the spatial IoU for each frame per class to get the frame average precision score and compute the spatio-temporal IoU per video per class to get the video average precision scor score. This is then averaged to obtain the f-mAP and v-mAP scores over various thresholds. The frame-AP reflects the average precision of detection at the frame level for each class, which is then averaged to obtain the f-mAP [133]. The video-AP reflects the average precision at the video level, which

is averaged to obtain the v-mAP score [133, 77]. For video segmentation, we use the $\mathcal{J}$ score for evaluation, which is given by the average IoU between the prediction and ground truth mask [4]. We also measure the average boundary similarity between prediction and ground truth as the $\mathcal{F}$ score [4].

4.2.5 Baseline methods

We explore several baselines to understand their limitations on video action detection. First, we use random and equidistant frame selection where random selection select the frames at random in each stage, equidistant uses equal distance between the frames during selection. Next, we extend existing AL methods used in image-based object detection [7, 8] to video action detection, where we score each frame using their algorithm for frame selection. We improve upon the uncertainty sampling for video level selection from [95] and compute uncertainty at pixel-level in all our baselines. We train all baselines using same action detection backbone for a fair comparison. We have random, equidistant, uncertainty-based [7] and entropy-based [8] approaches as baseline methods to compare against.

Table 4.1: Comparison between different baseline methods in UCF-101 and J-HMDB dataset for different frame annotation percent. * is extended to video action detection using same backbone detector network as ours.

	UCF-101						J-HMDB					
	f-mAP@0.5			v-mAP@0.5			f-mAP@0.5			v-mAP@0.5		
Method	1%	5%	10%	1%	5%	10%	3%	6%	9%	3%	6%	9%
Random	60.7	66.5	69.3	59.2	66.4	69.9	58.3	69.3	71.6	57.4	64.6	70.4
Equidistant	61.8	66.2	68.4	61.7	67.2	69.0	57.4	67.5	71.4	56.9	64.9	66.8
Gal et al.* [7]	60.9	66.7	68.9	59.4	66.8	69.1	58.2	66.7	67.5	57.4	66.8	67.4
Aghdam et al.* [8]	61.4	67.9	69.8	60.1	67.9	70.0	58.8	71.2	71.1	57.7	66.7	71.2
Our	**64.7**	**70.9**	**71.7**	**63.9**	**71.8**	**73.2**	**68.8**	**74.1**	**74.5**	**65.6**	**70.8**	**74.0**

4.2.6 Results

4.2.6.1 Comparison of baseline methods

We evaluate random, equidistant, entropy-based [8] and uncertainty-based [7] selection methods as baselines and compare with our approach in Table 4.1. While all baselines are effective for AL in image-based detection/classification tasks, we demonstrate that for video action detection prior methods [8, 7] perform similar or worse than random or equidistant methods. The lack of temporal information prohibits prior methods to select frames effectively as videos have sequential frames in same region with high uncertainty. Our approach accounts for the temporal continuity and outperforms all baselines including prior AL based methods [7, 8] consistently on both dataset for all annotation percent. This demonstrates that extending image-based methods is not well suited for video action detection task as shown in Figure 4.3.

4.2.6.2 Evaluation of proposed method

We evaluate our approach on UCF-101 and J-HMDB for action detection and compare with fully-supervised training in Table 4.2. For **UCF-101** we initialize with 1% of labelled frames and train the action detection model with a step size of 5% in each cycle. We achieve results very close to full annotations (v-mAP@0.5, 73.20 vs 75.12) using only 10% of annotated frames, which is a huge reduction (90%) in the annotation cost. For **J-HMDB**, we initialize with 3% labels as it is a relatively smaller dataset and it is challenging to train an initial model with just 1% labels. Here, we obtain results comparable with 100% annotations with only 9% of labels.

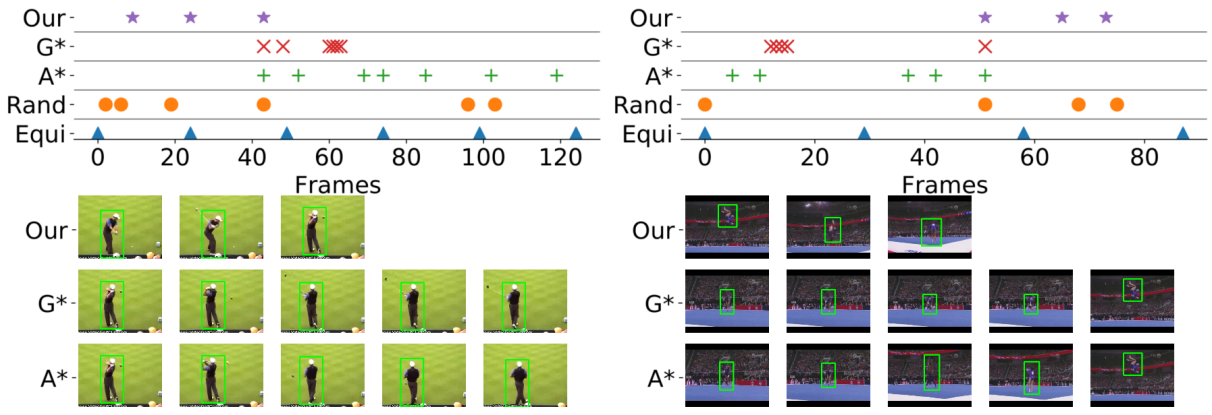

Figure 4.3: Analysis of frame selection using different methods. The x-axis represents all frames of the video, with each row representing a baseline method. The markers for each method mark the frames selected using that method. For both samples, our method selects distributed frames centered around action region, Gal et al [7] **[G*]** selects frame around same region since there is no distance measure and Aghdam et al. [8] **[A*]** selects slightly more distributed frames but theose are not from crucial action region. [G*:Gal et al[7], A*:Aghdam et al[8], Rand: Random, Equi: Equidistant]

Table 4.2: Evaluation of our proposed ASL method on **UCF-101** and **J-HMDB**. We show v-mAP and f-mAP scores at various thresholds for 1%, 5% and 10% annotation selected using our APU algorithm. The model is trained with sparse labels using our MGW-Loss.

UCF-101					**J-HMDB**				
Annot	v-mAP@		f-mAP@		Annot	v-mAP@		f-mAP@	
Percent	0.3	0.5	0.3	0.5	Percent	0.3	0.5	0.3	0.5
1%	89.01	63.94	83.85	64.69	3%	95.15	65.56	89.94	68.78
5%	90.95	71.89	88.71	70.91	6%	95.20	70.75	93.09	74.09
10%	91.12	73.20	88.72	71.75	9%	95.58	74.01	92.67	74.50
100%	91.49	75.15	89.08	74.02	100%	96.39	75.75	93.74	74.91

4.2.6.3 Comparison to prior weakly/semi-supervised approach

We compare to prior weakly/semi-supervised action detection approach [76, 74, 55, 53, 10, 56] in Table 4.3 and explain their limitations. [10] uses external human and instance detectors to

Table 4.3: Comparison with state-of-the-art methods. We evaluate our approach using v-mAP and f-mAP scores using only 10% annotations. 'Video' uses video-level class annotations and 'Partial' uses sparse temporal and spatial annotations. V: video labels, P: points, B: bounding box, O: off-the-shelf detector. **f@** denotes **f-mAP@**

Method	Annot					**UCF-101**					**J-HMDB**				
						f@	v-mAP@				f@	v-mAP@			
	Percent	V	P	B	O	0.5	0.1	0.2	0.3	0.5	0.5	0.1	0.2	0.3	0.5
Fully supervised															
Peng et al. [37]	100%					65.7	77.3	72.9	65.7	35.9	58.5	-	74.3	-	73.1
TCNN [23]	100%					67.3	77.9	73.1	69.4	-	61.3	-	78.4	-	-
Gu et al. [134]	100%					76.3	-	-	-	59.9	73.3	-	-	-	-
ACT [112]	100%					69.5	-	76.5	-	-	-	-	74.2	-	73.7
STEP [24]	100%					75.0	83.1	76.6	-	-	-	-	-	-	-
Rel. Graph [26]	100%					77.9	-	-	-	-	-	-	-	-	-
AIA [27]	100%					78.8	-	-	-	-	-	-	-	-	-
VidsCapsNet [5]	100%					78.6	98.6	97.1	93.7	80.3	64.6	98.4	95.1	89.1	61.9
Weakly/Semi-supervised															
Mettes et al. [53]	Video	✓			✓	-	-	37.4	-	-	-	-	-	-	-
Escorcia et al. [74]	Video	✓				-	-	45.5	-	-	-	-	-	-	-
Zhang et al. [75]	Video	✓			✓	30.4	62.1	45.5	-	17.3	65.9	81.5	77.3	-	50.8
Arnab et al. [54]	Video	✓			✓	-	-	61.7	-	35.0	-	-	-	-	-
Weinz. et al. [10]	Partial	✓		✓	✓	63.8	-	57.3	-	46.9	56.5	-	-	-	64.0
Mettes et al. [55]	Partial	✓	✓			-	-	41.8	-	-	-	-	-	-	-
Cheron et al. [76]	Partial	✓			✓	-	-	70.6	-	38.6	-	-	-	-	-
Kumar et al. [56]	20%	✓		✓		69.9	-	95.7	-	72.1	64.4	-	95.4	-	63.5
Ours	10%	✓		✓		71.7	98.1	96.5	91.1	73.2	74.5	99.2	98.4	95.6	74.0
Ours	100%					74.0	98.3	96.9	91.5	75.2	74.9	99.2	99.2	96.4	75.8

build tubes aligned with 1-5 random spatially annotated GT frames per tube. This incurs larger annotation cost without any frame selection metric while having low performance. [53, 55, 54] follow Multi Instance Learning (MIL) approach, where [53] uses off-the-shelf actor detectors to generate pseudo-annotations and [55] relies on user input for point annotation *for every frame*, requiring large annotation cost. [54] expands on MIL approach combined with tubelets generated by an off-the-shelf human detector. While MIL based approach requires less oversight, it also suffers from reduced performance, even with state-of-the-art detectors.

[74] uses actor detector with video-level label to perform action detection, using a less involved approach as [54], but both have high label noise and low performance. [56] uses consistency regularization to train with unlabeled data in semi-supervised fashion. [76] uses discriminative clustering instead of MIL to assign tubelets to action label with various level of supervision, [75] uses combination of different actor detectors to build tube to train with video labels. They rely on multiple off-the-shelf components to generate the tubelets and suffer from low performance. [75] and [10] report their J-HMDB results using bounding-box annotation instead of the fine-grained pixel-wise annotation due to their design limitation to use external bounding-box detector for tube generation. Our approach does not rely on such detectors and can work with both bounding-box (UCF-101) and pixel-wise (J-HMDB) annotation and is comparable to the supervised performance.

4.2.7 Ablations

4.2.7.1 Effect of loss function

We evaluate the effectiveness of *MGW-Loss* for video action detection with sparse labels and compare it with baseline masking and interpolation based loss in figure 4.4. The proposed *MGW-Loss* learns better in sparse label conditions due to the approximated ground truth frames from interpolation. Without the approximated frames, the formulation in Eq. 4.4 will reduce to masking loss as $\sigma \rightarrow 0$. Masking computes loss only on the sparse ground truth and does not perform as well as the *MGW-loss* with interpolated ground truth as seen in figure 4.4. Our Gaussian based interpolation adapts better for approximated labels compared with simple interpolation due to having different weight for each frame based on their distance from real ground truth annotation.

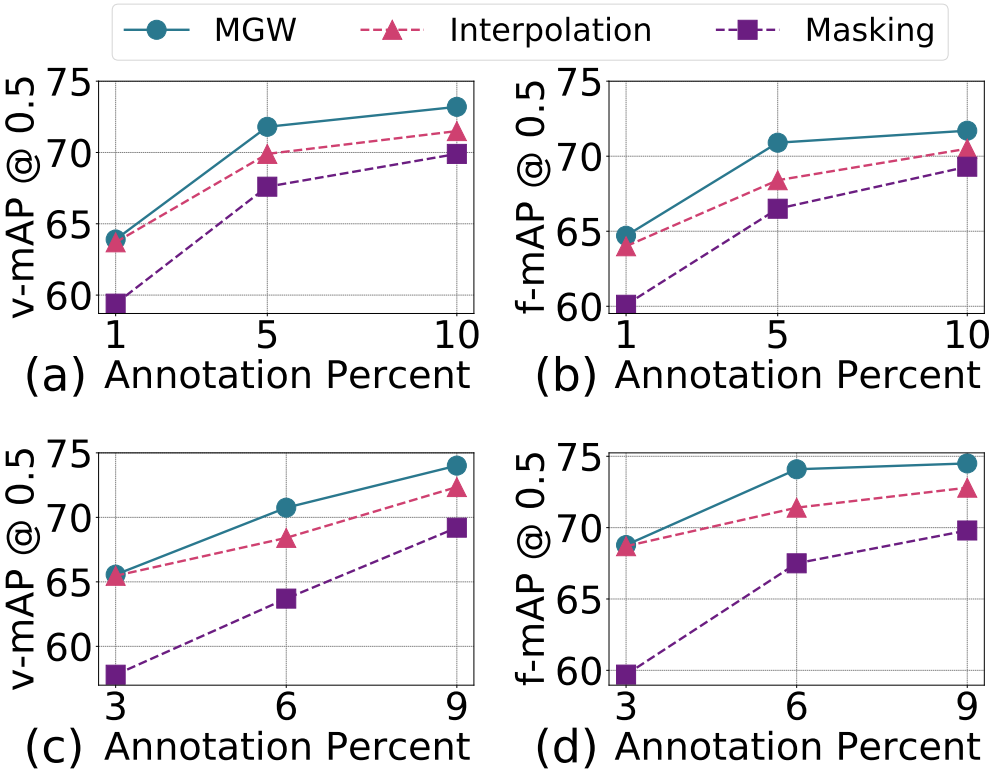

Figure 4.4: Comparison of different loss functions for UCF-101 (a-b) and J-HMDB (c-d). We report scores for v-mAP and f-mAP at IoU of 0.5 at various annotation percentages.

4.2.7.2 Effect of selection criteria

We compare how commonly used entropy and uncertainty-based selection methods perform against proposed *APU* algorithm when using the same loss formulation from Eq. 4.4. Figure 4.5 shows that *APU* has optimum frame selection as it encourages diverse selection by using adaptive distance to existing frames for the scoring process. Following [8], entropy based selection has a less effective fixed distance filter to avoid nearby frames. The uncertainty method lacks any distance component and performs worse than random or equidistant, selecting frames from nearby regions as seen in figure 4.3.

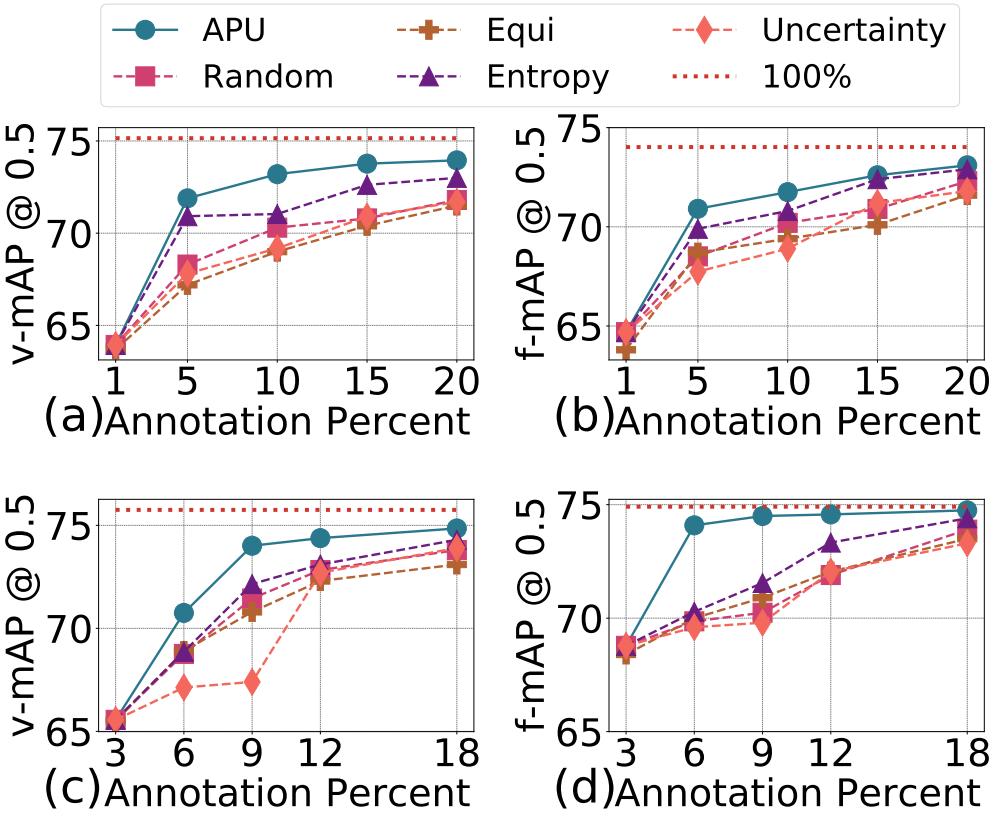

Figure 4.5: Comparison of different frame selection methods combined with MGW-Loss showing v-mAP and f-mAP scores for IoU @ 0.5 for (a-b) UCF-101 evaluation up to 20% (∼40k frames) data annotation. (c-d) J-HMDB evaluation up to 18% (∼3800 frames). Our *APU* approach gets better performance at a lower annotation percentage (lower annotation cost).

4.2.7.3 Annotating more frames

We also evaluate adding additional frames until the scores start to saturate, shown in figure 4.5. We see that for UCF-101 at 20% annotation (∼40k frames) with bounding box for each frame, all methods score close to each other and are near fully supervised 100% training. Similarly, we notice similar pattern for J-HMDB dataset evaluation 18% (∼3800 frames) with pixel level semantic ground truth for each frame, where the scores for different methods start to converge. This

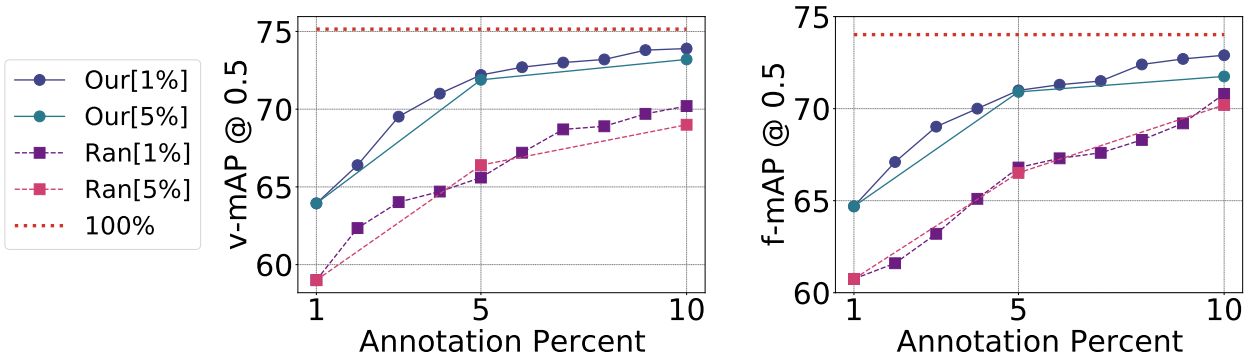

Figure 4.6: Comparison of our method with different annotation percent and step size against random selection method and fully-supervised method (100% annotations) on UCF-101. Results for step size increment of 1% and 5% are shown at 1%, 5%, and 10% annotations for our ASL based selection and random selection (denoted by *Ran*).

demonstrates that while the frame selection eventually converges with more data, our approach gets better score at an earlier stage, saving overall annotation cost.

4.3 Analysis

4.3.1 Variation in budget steps

Lower budget steps enables selection of fewer frames with high utility in each step instead of selecting more frames with low utility in higher budget steps. As the annotation set is more curated in each step in lower steps, we end up with better frames for the same annotation budget as higher steps. We evaluate the effect of using step size of 1% and 5% in figure 4.6 for UCF-101 dataset, starting from 1% till 10%. Step size of 1% has constantly better v-mAP and f-mAP score throughout, showing that smaller steps give greater performance. However, smaller step size needs more iterations, taking more time as a trade-off for better performance.

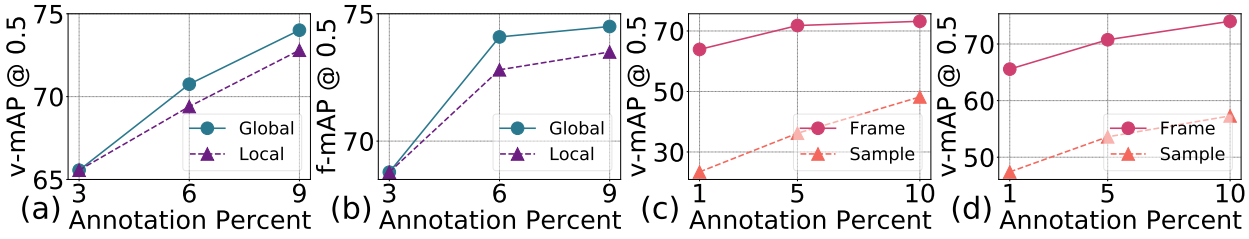

Figure 4.7: Analysis on sample selection strategy. (a-b)Global vs local frame selection strategy using *APU* on J-HMDB. (c-d) Frame vs sample selection for UCF-101 (c) and J-HMDB(d).

4.3.2 Local vs. global selection

The proposed approach is focused on sparse labeling where frames with high utility within a video are selected for annotations. However, it is important to note that videos as a whole have varying utility. To exploit this aspect, we explore two different frame selection strategies, local selection and global selection. In local selection, each video has a fixed budget b/N^v, where b is budget per cycle and N^v is the total number of videos in our training set. However, frames in global selection are taken from a global pool which includes frames from all videos, ranking based on overall dataset utility. Figure 4.7 (a-b) shows that global selection outperforms local selection strategy, emphasizing that some videos can be more informative than others as confirmed in figure 4.8.

4.3.3 Sample vs. frame selection

We follow [55] and annotate the entire sample (video) instead of finding the most useful frames within each sample. We compute pixel level uncertainty which is averaged over all the pixels in a frame using Eq. 4.1 and then averaged over all frames in a video to get the video level score. While this approach is simpler, it has higher cost during annotation with lower data variation. Let us assume a fixed cost of c per frame with f frames to annotate, we can assume a budget of

Figure 4.8: Sparse selection analysis. (a) Histogram showing number of frames selected per video using our method on UCF-101. Videos on the right show two samples from extreme ends of this histogram as marked in the plot. (b) Samples for cricket bowling class with APU selected frames on red marker (APU selects only two frames for 1 action instance). (c) Samples for salsa spin class (APU selects multiple frames (red) as each spin instance is visually diverse).

$B = c \times f$. We could distribute the frames across the set by picking only few important frames from each video, which would increase variation in the training set. However, if we annotate entire sample, there will be many redundant annotations with little gain, which is why frame selection performs better for action detection task as observed in figure 4.7 (c-d).

Table 4.4: Comparison of the proposed method on YouTube-VOS dataset with baseline AL methods using STCN [9]. *A* = Aghdam et al. [8], *G* = Gal et al. [7]. * is extended to video object segmentation using same network as ours.

	Overall			$\mathcal{J}_S$			$\mathcal{J}_U$			$\mathcal{F}_S$			$\mathcal{F}_U$		
Method	10%	20%	30%	10%	20%	30%	10%	20%	30%	10%	20%	30%	10%	20%	30%
Random	28.4	42.3	42.5	29.1	42.9	43.8	25.8	38.5	38.6	30.2	44.3	45.0	28.4	43.5	42.7
A * [8]	30.1	45.6	47.2	31.5	45.4	47.6	26.7	43.4	47.9	22.8	46.7	48.8	17.6	46.8	44.6
G * [7]	27.9	45.1	48.8	28.5	50.8	48.5	24.8	42.0	46.6	29.7	42.1	49.8	28.7	45.5	50.4
Our	**31.7**	**58.6**	**66.7**	**33.6**	**58.2**	**66.7**	**27.8**	**54.3**	**61.5**	**35.2**	**60.6**	**69.1**	**30.1**	**60.9**	**69.7**

4.3.4 Generalization beyond action detection

We test generalization of proposed cost and loss function for video object segmentation task on the YouTube-VOS 2019. Table 4.4 shows that our proposed selection approach gets better $\mathcal{J}$ and $\mathcal{F}$

scores for video segmentation task compared to baseline AL methods and random frame selection method.

4.4 Summary

In this chapter, we present active sparse labeling (ASL), a novel approach for label-efficient video action detection. The proposed approach uses an uncertainty based scoring mechanism for selecting informative and diverse set of frames for action detection. In addition, we also propose a simple yet effective loss formulation which can be used to train a model with sparse labels. The proposed approach is promising in saving annotation costs and we show that merely 10% of labels can achieve performance comparable to fully supervised methods. We further demonstrate the generalization capability of the proposed approach for video object segmentation.

CHAPTER 5: VIDEO ACTION DETECTION USING HYBRID ACTIVE LEARNING

The work in this chapter has been selected for publication in the following paper:

Aayush J. Rana and Yogesh S. Rawat. "Hybrid Active Learning via Deep Clustering for Video Action Detection." Accepted for publication in the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023.[135]

In the previous chapter, we demonstrated that an active learning based selection approach is effective for selecting frames to annotate that improve label-efficient video action detection. However, this approach assumes all videos have some frames annotated and only performs frame level selection. In this chapter, we extend that to also do video level selection, where we propose a method to score and compare videos among each other and select few videos for partial frame annotation. This means we only select informative videos and annotate a few informative frames from those videos, making better use of the annotation budget.

The rest of the chapter is organized as follows: Section 5.1 describes the proposed method and training objective, Section 5.2 explains the experiment setup and results of the proposed method, Section 5.3 does the analysis of various components of the method, and Section 5.4 contains the summary.

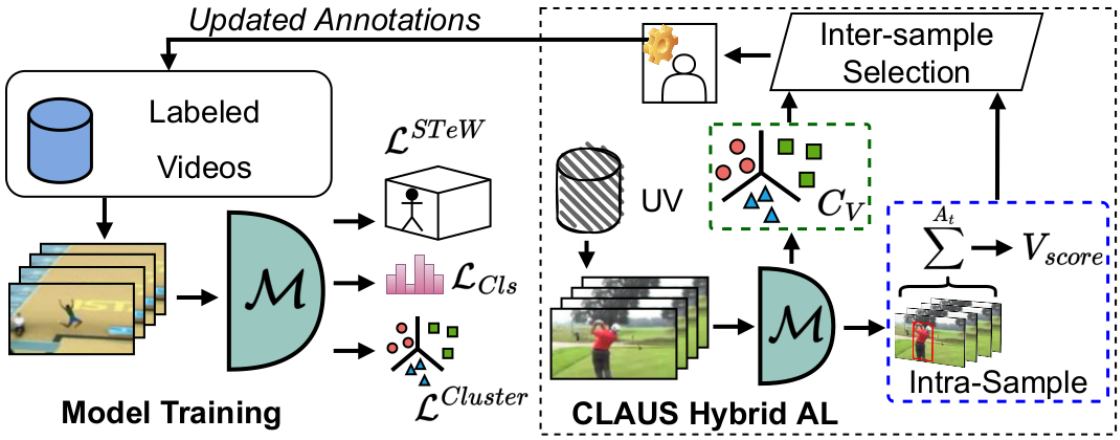

Figure 5.1: Overview of the proposed approach. Model training use videos with partial labels to learn action detection using the *STeW* loss and classification loss while also learning cluster assignment via cluster loss. The *CLAUS hybrid active learning* uses a trained model's output for intra sample selection and cluster assignment $\mathcal{C}_V$ for a video. Intra sample selection uses model score and selects top A_t frames of a video to get the video score (V_{score}). The V_{score} and $\mathcal{C}_V$ is used for inter sample selection and selected samples are sent to oracle for annotation. UV: Unlabeled videos.

5.1 Methodology

5.1.1 Problem formulation

Given a set of N videos $\mathcal{V} = \{v_1, v_2, ...v_N\}$ with $\mathcal{F}$ total frames, we intend to select a subset of videos $\mathcal{V}_s^T \subset \mathcal{V}$ with $\mathcal{F}_s^T$ total frames and annotate only $\mathcal{A}^\mathcal{T}\%$ of frames from the subset $\mathcal{V}_s^T$ based on the total annotation budget $\mathcal{B}$. At the end we will have a subset of videos $\mathcal{V}_s^T$ that has $\mathcal{F}_s^T = (\mathcal{F}_L^T, \mathcal{F}_U^T)$ frames, where $\mathcal{F}_L^T$ are annotated and $\mathcal{F}_U^T$ are unannotated frames. The proposed approach enables the use of partial spatio-temporal annotation, utilizing both $\mathcal{F}_L^T$ and $\mathcal{F}_U^T$ for model training. We begin with an initial set of videos $\mathcal{V}_s^0 \subset \mathcal{V}$ with $\mathcal{F}_s^0 = (\mathcal{F}_L^0, \mathcal{F}_U^0)$ frames where $\mathcal{A}\%$ ($\mathcal{F}_L^0$) of these are annotated. We train the action detection model $\mathcal{M}^0$ using $(\mathcal{V}_s^0, \mathcal{F}_s^0)$ and use this trained model to select additional videos and frames using proposed AL to obtain new annotations.

The proposed AL approach selects a diverse set of informative videos for annotation from $(\mathcal{V} - \mathcal{V}_s^0)$ which is added to $\mathcal{V}_s^0$ to obtain $\mathcal{V}_s^1$. Next we select $\mathcal{A}\%$ informative frames from the selected videos $\mathcal{V}_s^1$ for annotation. This iterative process is repeated till desired performance is met or the total budget is exhausted. An overview of the proposed approach is shown in Figure 5.1.

Video action detection Video action detection requires spatial localization of the activity in each frame with temporal consistency of the predicted action location throughout the video. Most of the existing methods involve complex multi-stage training with dense frame-level annotations [20, 24, 112], making iterative training challenging due to large resource requirement and dependency on good region proposals [23, 44]. In this work, we utilize a simple one-stage approach which has state-of-the-art performance on action detection task and can be efficiently trained end-to-end using a single GPU [5]. We rely on the optimized version proposed in [56] to further reduce model complexity and the model is trained using spread loss for classification and binary-cross entropy loss for action localization.

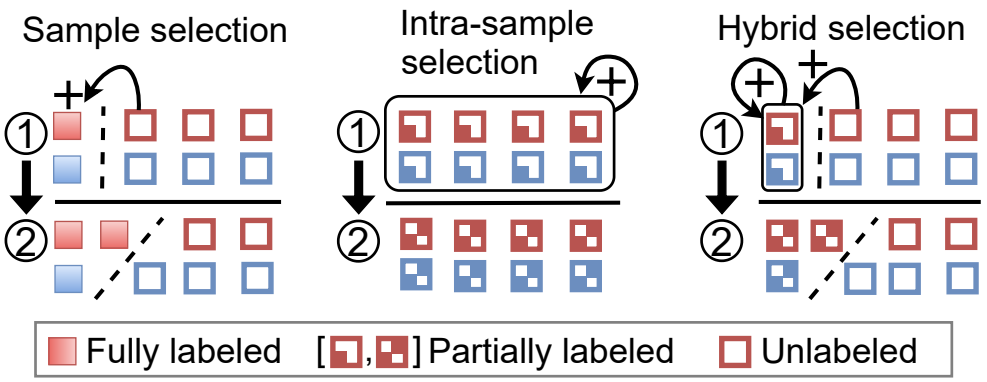

Figure 5.2: Overview of different active learning strategies for sample selection. We show a toy example for selection strategy as we add more annotatons to set 1 to obtain set 2. Sample selection approach takes unlabeled sample and annotates all frames in it. Intra-sample selects frames from all samples to annotate for the next set. Hybrid selects important samples and high utility frames to annotate for next set, significantly reducing overall annotation cost.

5.1.2 Hybrid active learning

The proposed hybrid active learning approach enables selection across unlabeled videos to identify diverse and important samples while also selecting limited frames within those samples for annotation; significantly reducing overall annotation cost. As shown in Figure 5.2, traditional sample selection approach simply selects and annotates entire sample, while intra-sample selection approach obtains frame-level annotations for all video samples. Sample selection does not take into account redundancy within a sample and intra-sample strategy on the other hand does not consider utility across samples and selects redundant samples, causing ineffective use of the annotation budget. We propose a hybrid approach that considers both intra-sample redundancy and inter-sample redundancy to select high utility frames and video samples. In addition, the proposed hybrid approach also integrates deep clustering to enable diversity along with informativeness while sample selection.

5.1.2.1 Inter-sample selection:

In active learning, several approaches have been studied to estimate informativeness of a sample [95, 98, 103]. Motivated by the recent success of uncertainty-based approaches [7, 83], we focus on model uncertainty [7] to predict the utility of a video sample. In case of classification task, video-level classification uncertainty can be sufficient, however, video action detection also requires localization of actions on every frame of a video. Therefore, spatio-temporal localization also plays a crucial role in estimating samples utility. To take this into account, we rely on spatio-temporal uncertainty in our approach.

We consider uncertainty in model's prediction at pixel-level to compute spatio-temporal uncertainty. We rely on MC-dropout to compute model uncertainty [7, 81, 67] as it is more efficient in

comparison with other approaches [8, 92, 89, 72]. The activity and non-activity region in a video will vary across action classes as well as across video samples. Therefore, uncertainty score based on all pixels in a video for sample utility will not be comparable across all unlabeled videos $\mathcal{V}_U$ for learning action detection. It will provide low uncertainty score for videos with short uncertain actions and long easy non-action regions which is not favorable for such videos. To overcome this issue, we propose to select limited frames in each video where we rank the video frames based on uncertainty and select the top $\mathcal{A}_t$ frames with high uncertainty. Given a pixel-level uncertainty $\mathcal{U}$, we compute the spatio-temporal uncertainty at video-level as,

$$V_{score} = \frac{1}{\mathcal{A}_t} \sum_{i=1}^{\mathcal{A}_t} \sum_{p=1}^{P} \mathcal{U}_{i,p} \tag{5.1}$$

where, $\mathcal{A}_t$ is the number of frames to select from each video in an AL iteration and P is the total pixels in each frame. The pixel-level uncertainty $\mathcal{U}$ is computed as,

$$\mathcal{U} = \frac{1}{R} \sum_{r=1}^{R} -log(\mathcal{M}(p, r)) \tag{5.2}$$

where, $\mathcal{M}(p)$ is the model's prediction of pixel p for each frame, averaged over R different runs. Uncertainty values for $\mathcal{M}(p)$ below certain threshold (definite background) is set to 0. In our preliminary experiments, we observed that sample level classification uncertainty does not provide significant improvement over spatio-temporal uncertainty for sample utility. Therefore, we only utilize spatio-temporal uncertainty in our approach to determine sample informativeness for action detection.

5.1.2.2 *Intra-sample selection:*

The informative videos selected in inter-sample selection $\mathcal{V}_{s-prime}^{t}$ are added to the existing set $\mathcal{V}_{s}^{t-1}$ to obtain $\mathcal{V}_{s}^{t}$. In intra-sample selection, we select frames with high utility from these videos $\mathcal{V}_{s-prime}^{t}$ for frame-level annotation. We rely on frame-level model uncertainty $U_f = \sum_{i}^{I}(\mathcal{U}_i)$ for all I pixels in a frame to estimate frame utility for action detection. Here $\mathcal{U}$ is pixel-level uncertainty as described in Equation 5.2. Since pixel-level uncertainty $\mathcal{U}$ is already computed for spatio-temporal uncertainty, intra-sample selection has no computation overhead.

5.1.2.3 *Diverse sample selection:*

Model uncertainty can be used for sample selection focusing on their informativeness. However, it does not ensure diversity among selected videos and there can be redundancy in such a selection strategy. A simple solution to address this issue can be developed with the help of class labels. However, this will require additional annotations which defeats the purpose of saving annotation cost. We propose an implicit clustering approach which utilize latent video features and does not require additional annotations. More specifically, we use deep clustering [136] which learns the cluster representation for each category from the known labeled subset $\mathcal{V}_{s}^{0}$ and adapts the clusters as the latent features of each video changes during training.

To enable diverse sample selection, we model the relation between diversity of each unlabeled sample $\mathcal{V}_U$ with already labeled samples $\mathcal{V}_L$. The proposed clustering approach allows the model $\mathcal{M}$ to learn latent features $\mathcal{LF}$ which represent each sample in a cluster. The objective of the model $\mathcal{M}$ is to improve the latent features such that it is close to the corresponding cluster center for that sample. The clustering objective is defined as,

$$\min_{\theta} \mathcal{L}^{Cluster} = \sum_{i=1}^{N} \frac{\lambda}{2} ||\mathcal{LF}(x_i|M_\theta) - C_K(x_i)||^2 \tag{5.3}$$

where, λ is a scaling term for the loss, θ is the parameters for model $\mathcal{M}$, $\mathcal{LF}$ is latent feature for sample x_i where $i \in [1, N]$ and C_K is the cluster center for sample x_i.

We first compute informativeness scores for each video in $\mathcal{V}_U$ using Equation 5.1, and then find cluster in $\mathcal{C} = [c_1, c_2, ...c_k]$ with K total clusters corresponding to each unlabeled video. The total number of videos to be selected in a cycle is constraint by current budget $\mathcal{B}_v$. We limit the samples selected per cluster such that the selection is proportional to the cluster size. For any cluster with n_c videos, we assign a budget of $n_c \times \mathcal{B}_v/N_U$, where N_U represents total number of unlabeled videos. The selection algorithm is further detailed in supplementary. We argue that nearby frames in a video will have similar model uncertainty and redundant utility. So, we avoid selecting nearby frames in intra-sample selection to ensure diversity while frame selection.

5.1.3 Training objective for partial label learning

Traditional video action detection method relies on actor annotation for each frame in order to train a model for action localization and classification [24, 5, 48]. However, in case of partial annotations it is not possible to train localization without annotations, which limits the use of these approaches directly. We propose a novel loss formulation which can effectively utilize partial annotations for localization.

Spatio-Temporal Weighted (STeW) loss The partial spatio-temporal annotations can be converted into dense pseudo-labels with the help of interpolation [10]. However, these pseudo-labels can have errors due to motion of actor/camera in a video and temporal gap between the partial labels. We propose to use temporal continuity of actions to mitigate this issue and enable effective utilization of partial annotations. We hypothesize that actions have some temporal continuity across time which may vary with different actions. By leveraging this temporal continuity in a video, we compute spatio-temporal weight for each pixel independently which captures the confi-

dence of a pseudo-label.

First, we compute the psuedo-labels using interpolation between the annotated frames as [10]. Next, we apply a spatio-temporal weight to suppress incorrect pseudo-labels. We compute the overlap of annotation for nearby frames and assign each pixel a weight based on the overall consistency which is given as,

$$\phi_f^{i,j} = Dist(f_a - f)\frac{1}{(W+1)}\sum_{w=f-W}^{f+W} f_w^{i,j}, \tag{5.4}$$

where, weight ϕ of frame f with $i \times j$ pixels is combination of distance of frame f from nearest annotated frame f_a and average value of pixel i, j of nearby W frames. Our hypothesis is that the background and foreground should be consistent for most of the frame, except for the moving actions. The average value of nearby W pixels will give consistency value for each pixel, where we assign a weight of 1 for consistent background/foreground ($\leq P_{low}$ or $\geq P_{high}$) and average value for other inconsistent pixels. The final localization loss with spatio-temporal weight is computed as,

$$\mathcal{L}_l^{STeW} = \frac{1}{F}\sum_{f=1}^{F} \phi_f L_l^f, \tag{5.5}$$

where, for a video with F frames, L_l^f is the BCE localization loss for f^{th} frame and $\phi_f \in [0, 1]$ is the pixel-wise spatio-temporal weighted mask from Equation 5.4 for f^{th} frame.

Overall training objective Our overall training objective is given as,

$$\min_{\theta} \mathcal{L} = \mathcal{L}^{Cluster} + \mathcal{L}_l^{STeW} + \mathcal{L}^{Cls} \tag{5.6}$$

where, θ is the model parameters, $\mathcal{L}^{Cluster}$ is cluster loss from Equation 5.3, $\mathcal{L}_l^{STeW}$ is detection loss from Equation 5.5 and $\mathcal{L}^{Cls}$ is the spread loss for classification from [5].

5.1.4 Sampling budget and cost incorporation

For each stage of annotation we assume a fixed budget $\mathcal{B}$ which will be separated to annotate video label and to annotate frames in that video, given as $\mathcal{B}_v$ and $\mathcal{B}_f$ respectively. Annotating each video label will require a cost C_v since the annotator has to watch and identify the class. Similarly, annotating each frame with bounding-box or pixel-wise labels will require a cost C_f. Thus, for each stage we can only annotate videos and frames so that $C_v^{total} \leq \mathcal{B}_v$ and $C_f^{total} \leq \mathcal{B}_f$.

5.2 Experiments and results

5.2.1 Datasets

We evaluate our approach on **UCF-101-24** [3] and **J-HMDB-21** [2] action detection datasets. UCF-101-24 consists of 24 different action categories with spatio-temporal bounding-box annotations for 3207 untrimmed videos. J-HMDB-21 dataset has 21 categories with pixel-level spatio-temporal annotations for 928 trimmed videos.

5.2.2 Evaluation metrics

We measure the standard frame-mAP and video-mAP scores for different thresholds to evaluate our model's action detection results following prior works [37]. The frame-mAP reflects the average precision of detection at the frame level for each class, which is then averaged to obtain the f-mAP [133]. The video-mAP reflects the average precision at the video level, which is averaged to obtain the v-mAP score

5.2.3 Implementation details

Active learning We initialize our training with a set of videos $\mathcal{V}_L^0$ with class label and $\mathcal{A}\%$ annotated frames within those videos selected at random. We use **K=5** centers for clustering (analysis on varying K in supplementary) and **R=10** forward passes per video. For each stage, we select $v\%$ videos for annotation based on budget $\mathcal{B}_v, \mathcal{B}_f$, where the videos are given class label and $\mathcal{A}\%$ of their frames are annotated and added to $\mathcal{V}_L^0$. We repeat this until total budget is exhausted or desired performance is achieved.

Training details All our experiments are performed using PyTorch [130] on a single Nvidia Quadro 5000 GPU. The scores are average of 3 different runs. We adapt the video action detection model from [5] and use 2D capsules and I3D encoder [6] following [56], with pretrained weights from the Charades dataset [131]. The network is trained using Adam optimizer [123] with learning rate $5e-4$ and batch size 8. $P_{low} = 0.1$ and $P_{high} = 0.9$ is set empirically. We use random crop and horizontal flip for video augmentation during training. Interpolation is done using linear point interpolation for bounding-box (UCF-101-24) and CyclicGen [132] for pixel-wise (JHMDB). We compute uncertainty based on dropout during inference following [7]. We don't perform any hyperparameter tuning and use same set of parameter settings for all our experiments on both the datasets.

5.2.4 Baseline methods

We compare the proposed approach with several baselines to demonstrate its effectiveness. We develop two non-parametric selection method using random and equidistant frame selection (both using random video selection). We also use prior AL methods for object detection in images as baselines. We use uncertainty-based AL [7] and entropy-based AL [8] for scoring each frame and

Table 5.1: Evaluation of the proposed method on **UCF-101-24** and **J-HMDB-21** for [v-mAP, f-mAP] @ 0.5 IoU. We increase the amount of samples and frames in each stage using the proposed approach and compare with fully-supervised approach. $\mathcal{A}\%$ is percent of annotated frames and $\mathcal{V}\%$ is percent of videos used to sample frames from.

		UCF-101-24				**J-HMDB-21**	
$\mathcal{A}\%$	$V\%$	**v-mAP**	**f-mAP**	$\mathcal{A}\%$	$V\%$	**v-mAP**	**f-mAP**
0.25	5	45.0	50.1	0.15	5	4.7	27.2
0.50	10	54.3	55.6	0.30	10	41.6	45.3
0.75	15	57.6	59.4	0.45	15	52.5	54.8
1.00	20	61.8	61.6	0.60	20	56.0	60.5
1.25	25	65.5	65.6	0.75	25	57.6	60.9
1.50	30	67.2	66.9	0.90	30	58.3	61.7
2.00	40	68.6	68.5	1.20	40	61.3	62.7
2.50	50	69.2	69.3	1.50	50	63.7	64.0
5.00	80	72.2	72.1	5.40	80	71.5	72.8
90	90	73.6	73.0	90	90	73.1	73.0
100	100	75.2	74.0	100	100	75.8	74.9

do sample selection. All baselines use same action detection backbone as ours.

5.2.5 Results

We show that our iterative AL approach is able to improve results in each step and use only a fraction of the annotations to perform close to fully-supervised approach with 90% annotations in Table 5.1. We also perform detailed comparisons with 4 baselines in Table 5.2. We further compare our approach with previous weakly-supervised action detection approaches on both the datasets in Table 5.3 and 5.4.

Table 5.2: Comparison of the proposed approach with various baseline methods. All baseline methods use same action detection backbone as ours. We show the v-mAP and f-mAP metrics at 0.5 IoU. † is modified for video action detection using public code. $\mathcal{A}\%$ is total annotations.

		UCF-101-24		**J-HMDB-21**	
Method	$\mathcal{A}\%$	**v-mAP**	**f-mAP**	**v-mAP**	**f-mAP**
Random	1%	52.6	54.1	36.6	42.1
Equi.	1%	53.3	55	38.1	43.5
Entropy [8] †	1%	52.2	53.5	40.7	49.0
Uncertainty [7] †	1%	44.0	46.7	46.0	47.9
Our	1%	61.8	61.6	58.6	61.9
Random	5%	67.5	67.3	69.3	70.1
Equi.	5%	67.2	67.0	70.0	70.4
Entropy [8] †	5%	71.3	70.2	70.7	70.8
Uncertainty [7] †	5%	69.7	68.2	69.0	69.3
Our	5%	72.2	72.1	71.3	72.7

5.2.5.1 Comparison with baselines

Table 5.2 shows comparison of our method with random, equidistant, entropy-based [8] and uncertainty-based [7] AL baselines for UCF-101-24 and J-HMDB-21. We report the f-mAP and v-mAP scores at 1% and 5% total annotations. Random and equidistant give an idea of non-parametric sample selection where the videos are selected at random and the frames are selected at random or equidistant. We notice that these baselines give lowest scores. Then we compare with other AL baselines using [8, 7]. Since these are image-based, they are not well suited for frame ranking in videos as reflected by their scores. [8] ignores nearest 5 frames for each selection, but this still does not work as well as proposed diverse selection. Since these prior AL baselines don't have notion of similarity/distance for videos, we see that random performs comparably. In contrast, our approach gives best performance, highlighting the impact of cluster based diverse sample selection.

Table 5.3: Comparison with state-of-the-art weakly-supervised methods on UCF-101-24. We evaluate our approach on v-mAP and f-mAP scores using only 1% and 5% total frame annotations. 'V' uses video-level annotations and 'P' uses a fraction of the mixed annotation. 'S' denotes SSL methods. We report [10] with their scores for 2 (1.1%) and 5 (2.8%) frames annotated per video.

Method	$\mathcal{A}\%$	**f-mAP@**	**v-mAP@**			
		0.5	0.1	0.2	0.3	0.5
Mettes et al. [53]	V	-	-	37.4	-	-
Escorcia et al. [74]	V	-	-	45.5	-	-
Zhang et al. [75]	V	30.4	62.1	45.5	-	17.3
Arnab et al. [54]	V	-	-	61.7	-	35.0
Mettes et al. [55]	P	-	-	41.8	-	-
Cheron et al. [76]	P	-	-	70.6	-	38.6
Weinzaepfel et al. [10]	1.1%	-	-	57.1	-	46.3
Weinzaepfel et al. [10]	2.8%	63.8	-	57.3	-	46.9
MixMatch [62]	S-20%	20.2	-	60.2	-	13.8
Pseudo-label [137]	S-20%	64.9	-	93.0	-	65.6
Co-SSD(CC) [63]	S-20%	65.3	-	93.7	-	67.5
Kumar et al. [56]	S-20%	69.9	-	95.7	-	72.1
Ours	1%	61.6	98.1	95.9	88.9	61.8
Ours	5%	72.1	98.1	96.1	91.2	72.2

Table 5.4: Comparison with state-of-the-art semi-supervised methods on J-HMDB-21 using only 1% and 5% total frames annotation. 'V' uses video-level class annotations. 'S' denotes SSL method. We report [10] with their scores for 2 (6%) and 5 (15%) frames annotated per video.

Method	$\mathcal{A}\%$	**f-mAP@**	**v-mAP@**			
		0.5	0.1	0.2	0.3	0.5
Zhang et al. [75]	V	65.9	81.5	77.3	-	50.8
Weinzaepfel et al. [10]	6%	50.7	-	-	-	58.5
Weinzaepfel et al. [10]	15%	56.5	-	-	-	64.0
MixMatch [62]	S-30%	7.5	-	46.2	-	5.8
Pseudo-label [137]	S-30%	57.4	-	90.1	-	57.4
Co-SSD(CC) [63]	S-30%	60.7	-	94.3	-	58.5
Kumar et al. [56]	S-30%	64.4	-	95.4	-	63.5
Ours	1%	61.9	99.0	96.8	91.5	58.6
Ours	5%	72.7	99.1	97.3	94.8	71.3

5.2.5.2 *Comparison with weakly supervised approach:*

Our cluster based video and frame selection approach selects limited samples and can also be compared with prior weakly supervised methods for video action detection. Prior weakly supervised methods rely on multiple instance learning [53, 55, 54] or instance learning [10], paired with off-the-shelf actor detector or user-generated points to create GT annotations for training. These rely on multiple external components or require user to annotate points in each frame, reducing their practical use. Some methods are less involved with built-in detector branch [74] but suffer from noisy annotations. [76] applies discriminative cluster approach to match generated actor tubes with video label with partially annotated frames. [75] combines multiple actor detectors to build stronger GT annotations, relying heavily on external components. Our approach doesn't rely on external detection components and uses simple iterative approach to select useful limited samples. This allows our method to be easily used for training. Table 5.3 and 5.4 shows comparative scores with prior weakly-supervised methods.

5.2.6 *Ablations*

Effect of clustering We evaluate the effect of clustering for video selection in our approach in Figure 5.4. The selection approach without clustering simply selects *top-k* videos for further annotation, which ends up selecting some similar samples as it does not take diversity into account. Clustering increases sample diversity, as seen in Figure 5.3, which improves overall performance compared to non-clustering selection for both datasets as shown in Figure 5.4.

Effectiveness of STeW loss To evaluate the effect of our proposed *STeW* loss, we train the action detection network using simple *frame* loss and *interpolation* loss for UCF-101-24 dataset. *Frame* loss only computes loss for the annotated frame and ignores the pseudo-labels while *interpolation*

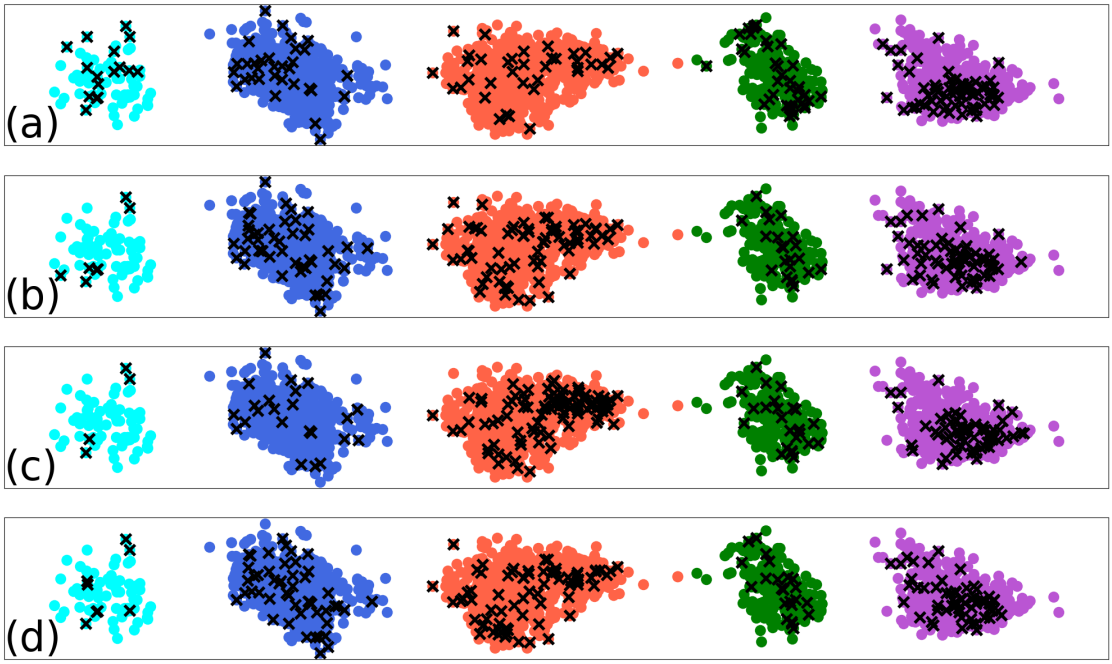

Figure 5.3: Visual representation of samples selected using (a) proposed *Clustering-Aware Uncertainty Scoring (CLAUS)*, (b) entropy, (c) uncertainty and (d) random selection methods using **x** marks. We get latent features of the videos from same iteration using same model and project them after PCA reduction. The clusters are from our clustering method and only for visual demonstration in (b), (c) and (d). We observe that our approach has diverse and even sample selection from different clusters while (b), (c) and (d) often selects samples closer to each other in terms of representation.

loss simply computes loss for all real and pseudo-labels equally. We use the same AL algorithm for all the approaches and show the result for UCF-101-24 for different steps in Figure 5.5. With less than 1% frames annotated, we see that *Frame* loss is not able to learn detection as well as *interpolation* and *STeW* loss. With the pseudo-labels created by interpolating the annotated frames, we see an increase in performance across all steps with both *interpolation* and *STeW* loss. Furthermore, the proposed *STeW* loss gives more importance to real frames and reduces the impact of the pseudo-labels that are inconsistent, performing best among all loss variations.

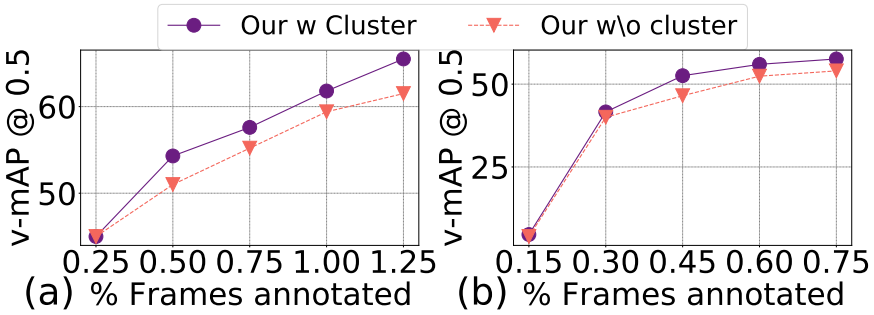

Figure 5.4: Comparison of our approach with and without clustering based selection for UCF-101-24(a) and J-HMDB-21(b).

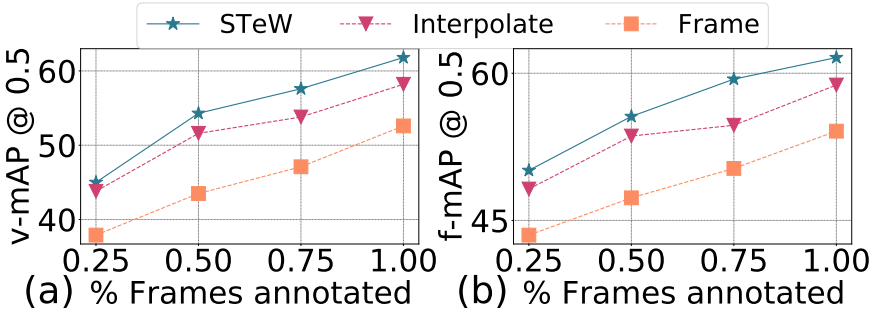

Figure 5.5: Comparison of proposed *STeW* loss with different loss variations combined with our ***CLAUS*** selection to train the video action detection network for UCF-101-24 dataset.

Effectiveness of CLAUS scoring We also evaluate different scoring functions (random, equidistant, entropy[8], uncertainty[7]) paired with the proposed *STeW* loss in Figure 5.6. Proposed *CLAUS* method is the only one that selects diverse samples based on global utility and is able to perform best compared to other scoring functions.

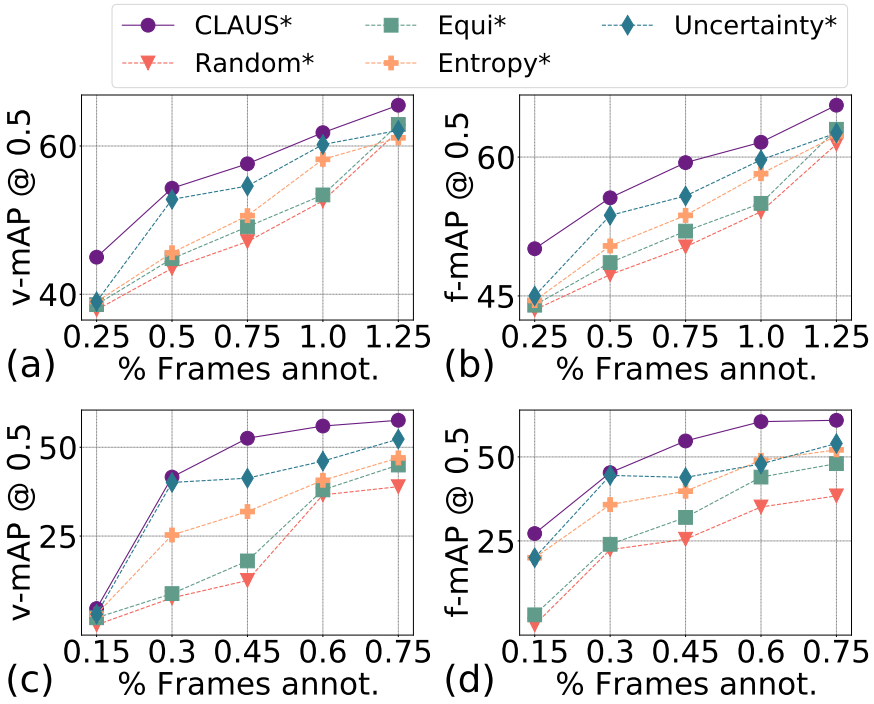

Figure 5.6: Evaluating various scoring methods for AL based annotation increments. * uses **our *STeW Loss*** for all selection approaches on UCF-101-24(a-b) and J-HMDB-21(c-d).

5.3 Analysis

Cost analysis Figure 5.7 compares cost to performance relation of our method and random selection. While having more annotation generally improves performance, our method selects diverse and important frames compared to random selection, resulting in significantly improved model in each step for the same cost. We further take the final model and evaluate per class performance for our and random selection in Figure 5.8. We outperform random selection for most classes while having fewer samples selected and give priority to select more samples for certain harder classes.

Class vs clustering diversity While samples from different classes add diversity, too many samples for easy classes will also add redundancy. Figure 5.8 shows that random approach has class

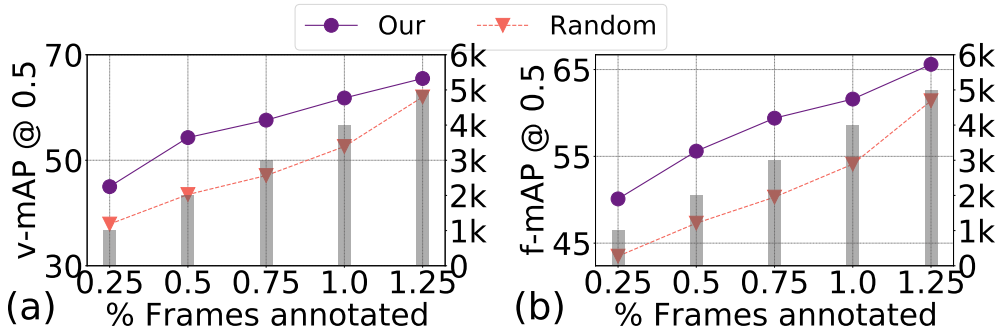

Figure 5.7: Performance evaluation of our method with random selection baseline on UCF-101-24 for various sample annotation percent. The cost of annotation for each step is shown by the shaded bars, with the cost value in the right axis in thousands.

balanced selection but performs below *CLAUS* as *CLAUS* reduces redundant samples from same class and prioritizes difficult and diverse samples.

Sample vs frame increment We evaluate effect of increasing only samples with a constant frame annotation rate of 5% and increasing both samples and frames annotation. Our goal is to get maximum performance gain with lowest cost. Increasing only samples with constant frame annotation rate has lower annotation cost than increasing both samples and frames for the same step. We show the results in Figure 5.10; having more training variation by adding only samples is more cost effective and has better performance than having more frames annotated for the same samples with higher cost. Interestingly, even random sampling that increases sample diversity performs better than our sampling with more frames, showing that sample diversity is an important factor in the selection process.

Selection strategy analysis We compare selection using proposed hybrid method against classical approaches for the same annotation budget. *Inter selection* assumes each video is fully annotated and randomly selects videos for given budget, thereby selecting fewer videos as more budget is used to annotate all frames. *Intra selection* assumes each video of the dataset is annotated for at

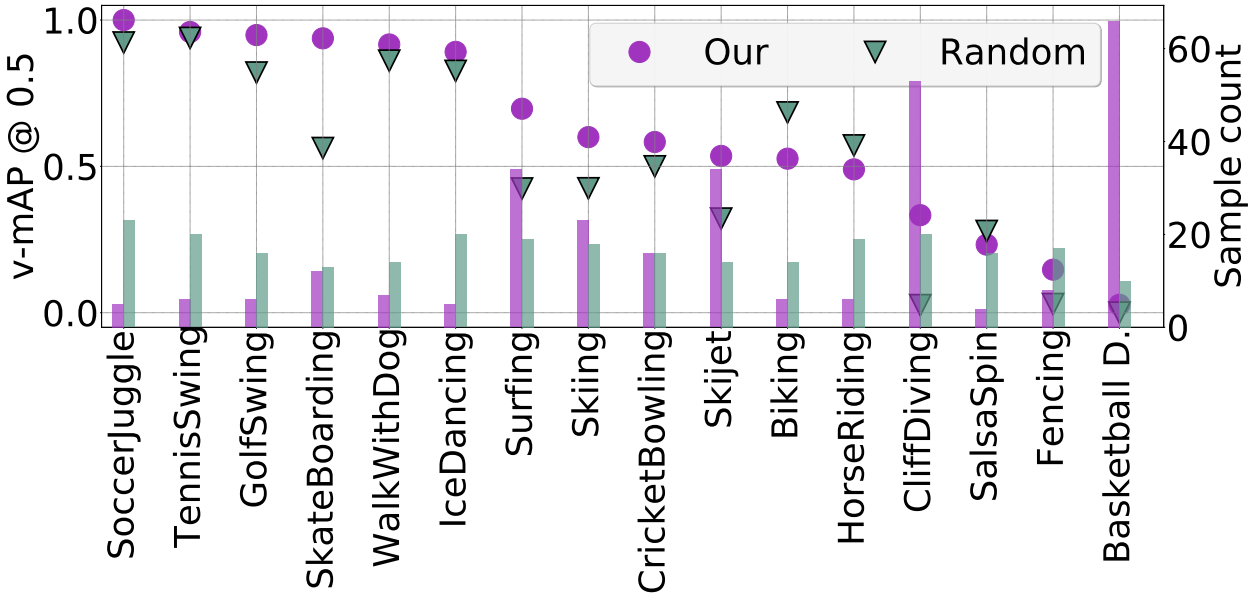

Figure 5.8: Analysis on performance across classes with varying amount of annotations. The scatter plot with markers on left axis shows v-mAP scores @ 0.5 IoU of our method against baseline random method on 16 action classes for UCF-101-24. The bar plot with markers on right axis shows per class sample distribution.

least 1 frame, spreading the budget over all videos. We show this comparison in Figure 5.9; our proposed method consistently scores better with both hybrid selection and random selection. Inter selection simply exhausts the budget in redundant frames from fewer videos and performs worst. Intra selection does perform close to our-with-random baseline due to larger sample variation.

Effect of number of clusters Our objective with cluster based sample grouping is to get a general representation of the sample which is not strictly based on the class label. This makes cluster assignment easier than having to identify 24 clusters for 24 classes in UCF-101-24 (21 in case of J-HMDB-21). We evaluate our proposed method using $K = 5$, $K = 10$ and $K = 15$ cluster centers on UCF-101-24 dataset. As shown in Figure 5.11, the overall performance is more or less close to each other for all values of K, while the performance for $K = 5$ is slightly better.

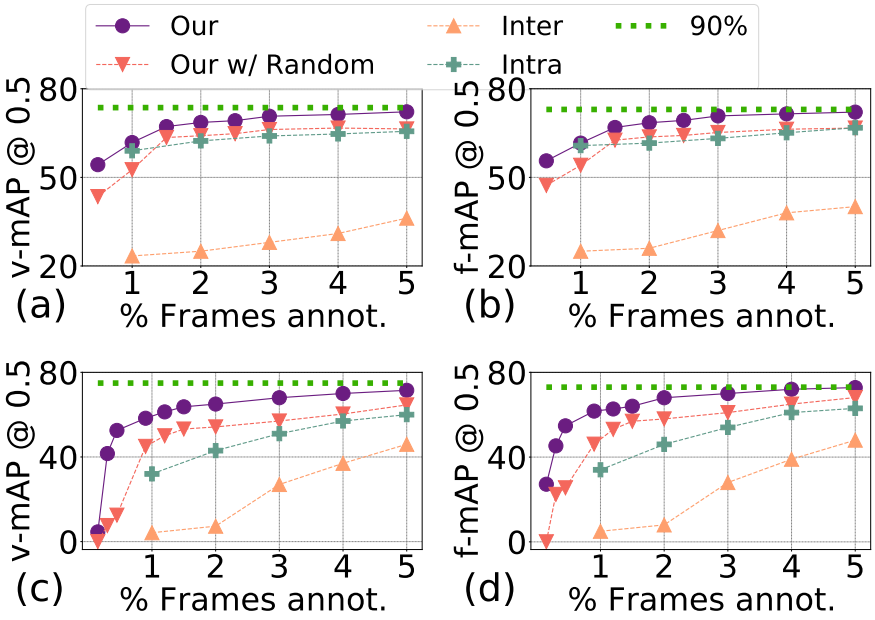

Figure 5.9: Comparison of the proposed CLAUS based AL method with random selection for video action detection. The plots show scores for (a-b) UCF-101-24 and (c-d) J-HMDB-21 for different annotation amount. The green line represents model performance with 90% annotations.

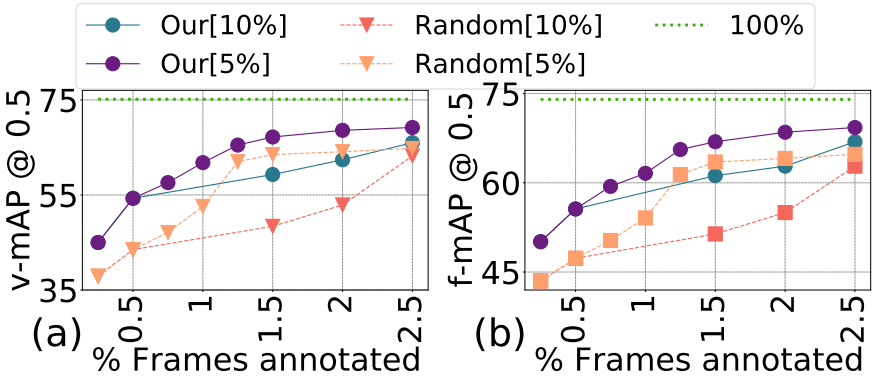

Figure 5.10: Performance difference for increasing sample and frame annotations [5%] vs increasing only frame annotations [10%] on UCF-101-24. Increasing both sample and frames at 5% increment adds diversity compared to only increasing frames, giving better scores.

Allowing the clusters to focus on features not tied to the classes and having fewer of such clusters performs better than having large cluster centers.

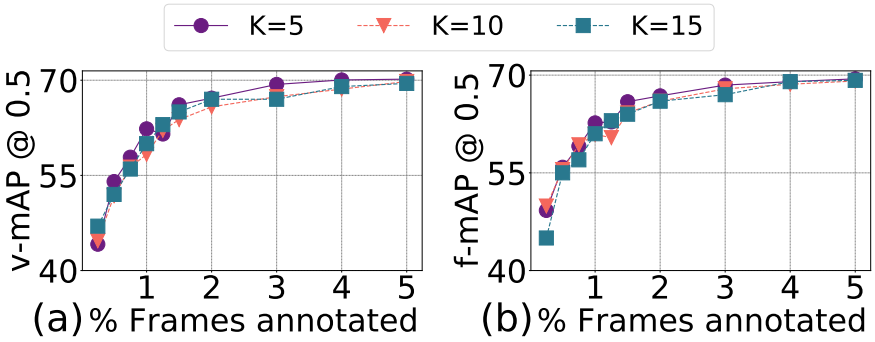

Figure 5.11: Comparison of model performance using different cluster centers (K). We train our proposed CLAUS using different cluster centers for UCF-101-24. We observe that model performance remains similar for different cluster center numbers.

5.4 Summary

In this chapter, we present a novel hybrid AL strategy for reducing annotation cost for video action detection. Our hybrid approach uses clustering-aware strategy to select informative and diverse samples to reduce sample redundancy while also doing intra-sample selection to reduce frame annotation redundancy. We also propose a novel *STeW loss* to help the model train with limited annotations, removing the need for dense annotations for video action detection. In contrast to traditional AL approach, our proposed hybrid approach adds more annotation diversity at the same cost. We evaluate the proposed approach on two different action detection datasets demonstrating its effectiveness in learning from limited labels with minimal trade-off on the performance.

CHAPTER 6: VIDEO ACTION DETECTION USING VARYING LEVEL OF SUPERVISION

The work in this chapter has been submitted as conference paper:

Aayush J. Rana and Yogesh S. Rawat. "OmViD: Omni-supervised active learning for video action detection" (submitted).

In this chapter, we focus on two different aspects of video action detection: 1) obtaining various types of annotations, and 2) using these omni-labels effectively for video action detection. We first explain the different types of annotations useful for video action detection. We then explain how we incorporate all the different annotation types into a unified training approach such that the model can learn action detection effectively. Lastly, we explain how to obtain additional data for each type of annotation such that it maximizes performance while keeping annotation cost low.

6.1 Methodology

6.1.1 OMNI-supervision

Omni-supervision training assumes presence of multiple types of annotations present in training set, with each sample having one or more annotation types present including tags, points, scribbles, boxes and masks as shown in figure 6.1. For a given set of training videos $\mathcal{V}$ with C classes, a sample $y_v = \{1...F_v\}$ with F_v total frames can have following annotation types:

Video-level tags only: $y_v = c$ where $c \in C$ classes. Such samples are only used to train the

classifier.

Point annotation: $y_v = c, p_i$ where $c \in C$ classes; $p_i = \{P_{x,y}\} \in \{Points\}_{i=1}^{F'}$ are point annotation of F' frames for the sample y_v with F_v total frames.

Box annotation: $y_v = c, b_i$ where $c \in C$ classes; $b_i = \{x_i^{min}, y_i^{min}, x_i^{max}, y_i^{max}\} \in \{Box\}_{i=1}^{F'}$ are bounding box annotations of F' frames for the sample y_v with F_v total frames. We don't assume the box annotation is present for all frames, so $F' < F_v$ and $|F'| \geq 1$.

Pixel-wise mask annotation: $y_v = c, m_i$ where $c \in C$ classes; $m_i = \{P_{x,y}\} \in \{Mask\}_{i=1}^{F'}$ are pixel-level mask annotation for pixels P of F' frames of sample y_v with F_v total frames (assuming $F' < F_v$ and $|F'| \geq 1$).

Scribble annotation: $y_v = c, s_i$ where $c \in C$ classes; $s_i = \{P_{x,y}^s\} \in \{Scrib\}_{i=1}^{F'}$ are pixel-level scribble annotation for pixels P^s of F' frames for the sample y_v with F_v total frames. Compared to pixel-wise mask annotation, the set of pixels P^s is smaller than P in mask annotation as scribbles only have pixels of a thin line drawn as scribble. We assume $F' < F_v$ and $|F'| \geq 1$.

Figure 6.1: Various types of annotations used for omni-supervised video action detection training. We show two frames from a video sample where each video is annotated with different annotation type.

6.1.2 OmViD: Video action detection

We propose **OmViD** as a simplified and unified approach to handle realistic variations in annotations for video action detection task. Given a set of training videos $\mathcal{V}$, we assume it has been separated into unlabeled set $\mathcal{D}_U$ and partially labeled set $\mathcal{D}_{PL}$. We use the partially labeled set $\mathcal{D}_{PL}$ which has $B\%$ box (mask) annotation, $S\%$ scribble annotation and $T\%$ video-level tag only annotations to train a unified model $\mathcal{M}$ such that we get prediction $\hat{y} = \mathcal{M}(v; \theta)$ for a given video $v \in \mathcal{D}_{PL}$; where θ is the trained model weight and prediction $\hat{y} = \{c, Det\}$ for $c \in C$ classes and spatio-temporal detection Det. For samples $t \in T$ with video-level tag only annotations, we assume there are no spatio-temporal annotations present and we only know the video label. However, for samples with box/mask/scribble annotation we assume that we know the video label as well as have at least one spatio-temporal annotation. Furthermore, we also don't restrict the sample to have only one type of spatio-temporal annotation, i.e. a single sample can have some frames annotated with box and some frames annotated with scribbles. Our experiments show that based on the level of utility for different frames in a sample, it is cost effective to have mixed annotation in a single sample.

Pseudo-label generation Superpixels group regions of a given frame F based on similarity in appearance. This property can be leveraged to extend sparse annotations and fill relevant regions of F to generate pseudo-labels for training. With sparse annotation in F' frames for a video with F_v frames such that $F' < F_v$, we will also have to fill connected regions spatially and temporally. For this, we propose using 3D superpixels to identify object regions overlapping the sparse annotation spatio-temporally. We identify the 3D superpixels that overlap with the sparse annotation and use all such regions as pseudo ground truth labels. These overlapping regions represent some portion of the object and can give a refined boundary for training purpose. This formulation can learn the 3D superpixel representations from the training data itself in an unsupervised manner and can be

jointly trained with the action detection model.

6.1.3 Learning objective

The proposed **OmViD** is train with mixed type of annotation at the same time; optimizing loss for classification and detection task. Given a video v from the partially labeled set $\mathcal{D}_{PL}$, we optimize the classifier loss $\mathcal{L}_{Cls}$ for all samples as we assume they at least have video-level tags annotation. As we add different types of spatio-temporal annotations, the training process for detection becomes complex as we have to account for pseudo-labels based on the type of annotation. We omit point annotation based on our preliminary experiments. We explain in detail about the different losses we use for such unified training.

6.2 Classification loss

We follow [5] for the class level classification and use spread loss for a given video. The objective function is defined as,

$$\mathcal{L}_{Cls} = \sum_{i \neq t} max(0, m - (a_t - a_i))^2 \tag{6.1}$$

where, a_t and a_i are the final class capsule's activation value for the target class and i^{th} class respectively. The margin is set as $m \in (0, 1)$ and is linearly increased from 0.2 to 0.9 during training cycle.

6.2.0.1 Detection loss

Since we assume the samples are not densely annotated, we leverage interpolation to get pseudo-labels and use weighted loss to assign appropriate value for each pseudo-label based on their reli-

ability. For a given video v with F_v total frames and F' annotated frames such that $F' < F_v$, we first generate pseudo-labels for neighboring frames without spatio-temporal annotations. We give higher weight for pseudo-labels closer to actual ground truth frame and lower weight for those further away. We generalize the detection loss for each sample as,

$$\mathcal{L}_{Det} = \sum_{i=1}^{F_v} W_i \mathcal{L}_i \tag{6.2}$$

where, for a sample v with F_v total frames, W_i is the weight of frame i based on closeness to real ground truth and $\mathcal{L}_i$ is the localization loss for frame i. For each spatio-temporal annotation type, we compute the loss independently and aggregate them all together. This simplifies the pseudo-label generation as well for each annotation type, specially if a single sample v has multiple annotation type in different frames. Next, we expand the detection loss for each type of annotation and explain the pseudo-label generation process.

Bounding box loss: For a sample v with box annotations $b_i = \{x_i^{min}, y_i^{min}, x_i^{max}, y_i^{max}\} \in \{Box\}_{i=1}^{F'}$ for F' frames, we generate pseudo-labels for unannotated frames F" such that $F_v = F' \cup F$". As fine pixel-level boundary is not needed, we use linear interpolation of boxes to get pseudo-labels. The box detection loss is given as,

$$\mathcal{L}_{Det}^{Box} = \sum_{i=1}^{F_v} W_i^{Box} \mathcal{L}_i^{Box} \tag{6.3}$$

where, W_i^{Box} is the weight for frame i based on distance from the closest real box annotation in $\{Box\}_{i=1}^{F'}$, and $\mathcal{L}_i^{Box}$ is the localization loss of the bounding box in frame i.

Pixel-wise mask loss: For sample with pixel-wise mask annotations, we cannot perform linear interpolation for pseudo-label generation of the mask. Instead, we can compute the pseudo-masks

prior to training using pixel-level interpolation. We can then compute the mask loss as,

$$\mathcal{L}_{Det}^{Pixel} = \sum_{i=1}^{F_v} W_i^{Pixel} \mathcal{L}_i^{Pixel} \tag{6.4}$$

where, W_i^{Pixel} is the weight for frame i based on closest real mask annotation and $\mathcal{L}_i^{Pixel}$ is the localization loss of predicted mask for i^{th} frame.

Scribble loss: The last variation we have is for scribble annotation, where for a given video v we only have scribble annotation $s_i = \{P_{x,y}^s\} \in \{Scrib\}_{i=1}^{F'}$ for frames F', where P^s are pixels with scribbles. We use 3D superpixels to get J total superpixels such that $[\exists s_i \{SPix_j(s_i)\}_{j=0}^{J}]$; where each superpixel $SPix_j$ has at least one pixel s_i from the scribble in it. This extends the scribble to spatio-temporally connected regions to generate pseudo-labels for detection training. This allows us to use sparse scribbles to compute the detection loss as,

$$\mathcal{L}_{Det}^{Scribble} = \sum_{i=1}^{F_v} W_i^{Scribble} \mathcal{L}_i^{Scribble} \tag{6.5}$$

where, $W_i^{Scribble}$ is the weight for frame i based on closest scribble frame and $\mathcal{L}_i^{Scribble}$ is the localization loss of prediction for i^{th} frame. Next, we describe the superpixel training process which enables pseudo-label generation for scribbles.

Superpixel loss:

Since scribbles are only thin pixels and don't tell us the exact area of the action region in a frame, traditional approaches (interpolation, largest box region) are not able to adequately address the problem of finding right areas to label. This leads us to learn superpixel segmentation to predict connected areas based on their features following [138, 139]. We extend [114] to predict 3D spatio-temporal superpixels and optimize the superpixel loss. First we extend the association map Q for each pixel to predict pixel property $f(p)$, where the property $f(p)$ is a 9-dimensional CIELAB

color vector representing a 3D spatio-temporal association map for pixel $p = [x, y, z]$ in a volume. The 3D superpixel loss is given as,

$$\mathcal{L}_{SLIC}(Q) = \sum_{p} \|f_{col}(p) - f'_{col}(p)\|_2 + \frac{m}{S}\|p - p'\|_2 \tag{6.6}$$

where, $f_{col}(p)$ is the 9-dimensional pixel property of pixel p, $f'_{col}(p)$ is the predicted property for pixel p, S is superpixel sampling interval, m is a weight balance, p is the location of pixel in a volume and p' is the predicted location. The first term encourages grouping of pixels with similar interest while the second term enforces spatial compactness of superpixels.

6.2.0.2 Full training objective

Based on the individual loss for each annotation type present and the superpixel training, our overall training objective can be specified as,

$$\mathcal{L} = \mathcal{L}_{Cls} + \lambda_B \mathcal{L}_{Det}^{Box} + \lambda_P \mathcal{L}_{Det}^{Pixel} + \lambda_S \mathcal{L}_{Det}^{Scribble} + \mathcal{L}_{SLIC} \tag{6.7}$$

where, $(\lambda_B, \lambda_P, \lambda_S) \in [0, 1]$ are weights that enable the specific loss if a given sample has corresponding annotation.

6.2.1 Sample selection

We start model training using an initial partially labeled set $\mathcal{D}_{PL}^1$. Once the model is trained, we want to increase overall videos by adding some high value samples from the unlabeled set $\mathcal{D}_U$ to get $\mathcal{D}_{PL}^2$. However, there is no specification to the type of annotation each new sample should get. We assume that there is a fixed budget for each round of annotation increment, such that we add $B'\%, S'\%, T'\%$ annotation to the original $B\%, S\%, T\%$ for box, scribble and video-level tag only respectively. We select new videos from $\mathcal{D}_U$ to get $\mathcal{D}_{PL}^2$. We also rank the videos in $\mathcal{D}_{PL}^2$

to select the annotation type for each video and select the frames to annotate. This selection can be done randomly or using a metric based selection approach. We end up with $\mathcal{D}^2_{PL}$ consisting of $B + B'\%, S + S'\%, T + T'\%$ annotation after one round of AL based sample selection. Next, we describe the AL strategy in detail.

6.2.1.1 AL based selection strategy

We use AL to select new videos from $\mathcal{D}_U$ and rank the videos in $\mathcal{D}^2_{PL}$ to select annotation type for labeling.

Uncertainty sampling: We follow prior AL methods and use uncertainty based sampling [7, 140] to score and select new samples and frames for annotation. We use the model's uncertainty in prediction to score frames, which can be used to get video level scores. Since we get spatio-temporal prediction for each frame, we compute frame-wise uncertainty given as,

$$\mathcal{U}_f = \sum_{i,j=0}^{I,J} U_{i,j} \tag{6.8}$$

where, $\mathcal{U}_f$ is the uncertainty for a frame f with $I \times J$ pixels and $U_{i,j}$ is the uncertainty of model prediction for i, j^{th} pixel. We average score of all frames to get the video level scores. Next, we rank the videos from highest to lowest and select a set V_s of top-k videos from unlabeled set $\mathcal{D}_U$ to add to $\mathcal{D}^1_{PL}$. This is determined by the budget for each round.

Annotation type selection: Once we have V_s, we combine them with prior partially labeled set $\mathcal{D}^1_{PL}$ to get $\mathcal{D}^{2'}_{PL}$. However, we don't need to fully annotate the new samples from V_s, or even give them the same type of annotation. In order to reduce annotation cost and extract maximum utility we further identify annotation type of each sample based on their rank. First we use the already computed video level scores from Equation 6.8, we rank all videos in $\mathcal{D}^{2'}_{PL}$. Then we simply take

top-b% videos to add $b\%$ box/mask annotations, then take next top-s% videos to add $s\%$ scribble annotation and finally take the remaining videos to annotate them with video-level tag. The top-b% and top-s% is fixed based on annotation budget.

6.3 Experiments and results

6.3.1 Datasets

We evaluate the proposed approach on UCF-101 [3] and J-HMDB [2] dataset for video action detection. UCF-101 contains 3207 untrimmed videos with spatio-temporal bounding box annotation for 24 action classes. J-HMDB has 928 trimmed videos with pixel-level spatio-temporal annotations for 21 action classes.

6.3.2 Evaluation metrics

Following prior works, we compute f-mAP and v-mAP at various thresholds to evaluate the performance on UCF-101 and J-HMDB datasets. We compute the average precision of detections for each class at the frame and video level, which is then averaged to obtain the f-mAP and v-mAP respectively [133, 77]. The video-mAP score is used to evaluate the detection's average precision at video level while the frame-mAP score is used to evaluate the average precision at frame level. These scores are computer per class and averaged to obtain the v-mAP and f-mAP scores.

6.3.3 Implementation details

Training details We define the proposed network architecture for training in figure 6.2. For a given video V with mixed annotation types, we pass it through a encoder-decoder network to predict the

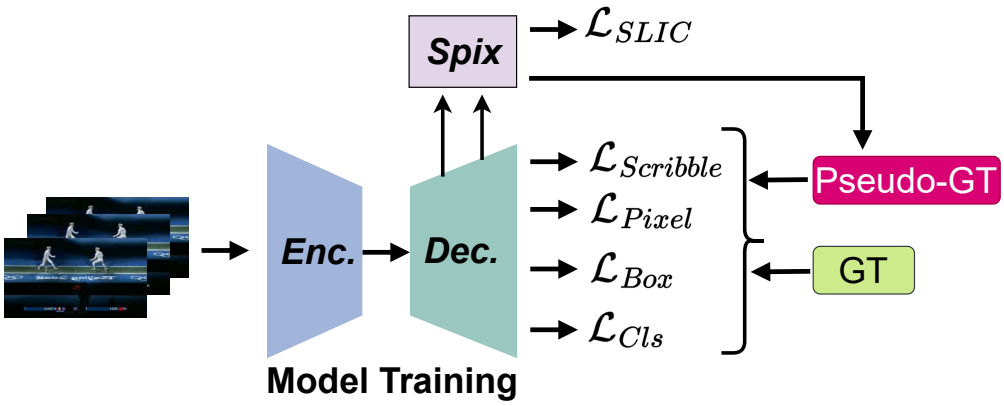

Figure 6.2: Our proposed network architecture used for training with mixed annotations. We use an encoder (**Enc.**) and decoder (**Dec.**) architecture following [5] for video action detection training. We train each video with all available annotation types including video tags, bounding box, pixel-wise mask and scribbles using the corresponding loss. We also propose a 3D superpixel prediction branch (**Spix**) which is trained in unsupervised fashion using the SLIC loss. We use combination of pseudo-GT (Ground Truth) and real GT to train the model for each annotation type.

action class and corresponding actor bounding box / pixel-wise mask. We follow [25, 56] for the action detection framework, which uses a 2D capsule variant of Video Capsule Network [5] with I3D [6] encoder using pretrained weights from Charades dataset [131]. The action prediction output from the decoder branch is used to then compute the loss of bounding box, pixel-wise mask and scribbles. We compute both bounding box and pixel-wise mask loss for each pixel of the output video, making the loss a binary classification problem (whether an action is present or not in the given pixel). We train the model using a single 16-GB GPU with a batch size of 6 for clips of dimension $8 \times 224 \times 224 \times 3$ as $depth \times height \times width \times channels$ as input for the network.

We also have a 3D superpixel branch (**Spix**), which takes the features from intermediate layers of decoder to do 3D superpixel prediction for each pixel of the video. The Spix branch follows the decoder branch structure and performs upsampling, 3D transpose convolution and skip connections for multiple layers. The Spix branch outputs a video where each pixel has the likelihood values for

all neighbors. Each pixel will have 27 neighbor values which indicates the superpixel it belongs to spatio-temporally. This is trained using the SLIC loss computed using ground truth superpixel for each pixel obtained from CIE-Lab version of the same video following [114].

Once we have the superpixels, we use that information to create pseudo-labels from scribble annotations. We also use interpolation for box/mask ground truth to get pseudo-labels for intermediate frames. We combine the pseudo-label ground truth and real ground truth to compute loss for all annotation types.

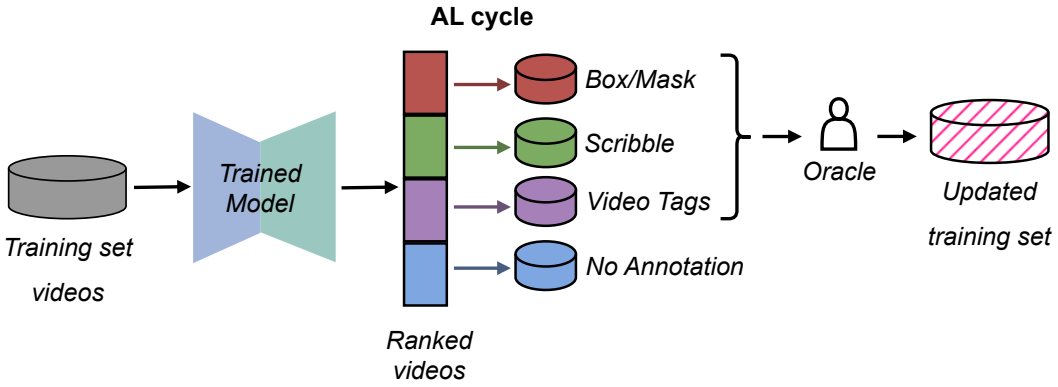

Figure 6.3: Our proposed active learning (AL) based sample selection strategy. For each AL cycle, we use a trained model from prior round to rank all training videos. Based on the ranking, we separate the videos into groups for different type of annotation to add. The selected videos for corresponding annotation type are sent to the oracle and added to the existing training set for future model training.

Active learning We use AL based sample selection to pick frames from videos for future annotation. As shown in figure 6.3, once we get a trained model we use it to score the entire training set. We follow [25] and use MC dropout in the model during inference and do 10 forward pass for each video to get the uncertainty in prediction. We use uncertainty score per frame to compute a video level score. Once we normalize the score for all videos, we rank them based on their overall uncertainty scores. The highest scoring videos are selected for box/mask annotation as they are the

most uncertain videos and the model could use highest quality annotations for such videos. Next group of videos are selected for scribble annotation as the model could use low cost and low quality annotation to learn from those based on their medium uncertainty score. Next we pick videos with low uncertainty for video level tags. Lastly, we don't annotate videos with very low uncertainty scores. During different active learning cycles the videos shuffle up and down the rank based on their changing uncertainty scores, so we get mixed annotations for videos which move to different annotation groups. After the oracle annotates each video's selected frames for the suggested annotation type, we add it to the updated training set.

Incremental supervision We use a small initial set of sparse annotations (bounding-box, mask) selected randomly from a small set of videos (20% for UCF-101, 30% for J-HMDB) with mixed annotation types equally divided among the videos. We only annotate 5 frames per video randomly for bbox/mask and scribble annotations, while videos with only tags have no frames annotated. During model training, we use 3D superpixels to generate pseudo-labels from the scribbles which is then used to train the detector. Videos with only tag annotation only get trained on classification and 3D superpixel loss. We perform annotation increment using the proposed AL selection strategy. For each round of annotation increment, we follow baseline approach and select new frames for annotation equally divided from a 20% subset of the training set. This subset is selected using AL based ranking and can contain new unlabeled videos and videos from prior rounds.

6.3.4 Cost overview

Annotation cost We standardize the cost of annotation for each type based on prior studies in order to get overall annotation cost. For each annotation type, we provide an estimate of annotation cost as follow:

- **Video level tags:** For videos with only 1 class per sample, we estimate the time to annotate as 3 seconds. This includes watching the video sample and putting a label to the video. For dataset with multiple labels per class, it takes additional 1 second per class to annotate [105, 104, 141].
- **Scribbles:** Based on [139], annotating every instance with scribble takes 10.9 seconds.
- **Bounding box:** Each high quality and accurate bounding box instance takes 35 seconds to annotate following [142] and [104].
- **Masks:** Annotating an accurate mask instance takes an average of 79.2 seconds based on [141].

Based on these annotation cost estimates, annotating training bounding boxes in UCF-101 for 24 classes takes **4,686 man hours**. Similarly, annotating J-HMDB for 21 classes with pixel-wise masks will take **487 man hours**.

Computational cost We compute the computation cost for the proposed uncertainty based active learning approach. We have to score the entire training set to get the rank for each video which is used to determine the type of annotations to select. Each round of AL selection takes about **1 hour** for UCF-101 with 2,284 training videos consisting of about 482,000 frames.

6.3.5 Baseline methods

We use the proposed **OmViD** approach and increment via AL to achieve action detection performance similar with fully-supervised approach. We evaluate a baseline using OmViD with random increment for comparison.

6.3.6 Results

We evaluate the performance of each dataset trained using various types of annotations mixed at different ratio. We first compare the performance of proposed **OmViD** method with prior weakly supervised action detection methods and demonstrate the performance for each method along with their annotation properties. Then we evaluate our method at varying levels of supervision to demonstrate the improvement in performance at different level of supervision.

Table 6.1: Comparison with state-of-the-art weakly-supervised methods using various annotation type and amount for UCF-101. We show the percent of annotation used as $\mathcal{A}\%$ for available methods and the cost of annotations as $\mathcal{C}\%$. $\mathcal{O}$ denotes if the method uses off-the-shelf object detector to generate training annotations. The annotation types used in training is denoted as: B$\rightarrow$ Box, P$\rightarrow$ Points, S$\rightarrow$ Scribbles, V$\rightarrow$ Video tag. S-20 use semi-supervised approach with 20% data. We compare f-mAP and v-mAP at different thresholds. We report [10] with their scores for 2 (1.1%) and 5 (2.8%) frames annotated per video.

Method	$\mathcal{A}_B\%$	$\mathcal{C}$	$\mathcal{O}$	Type				**f-mAP@**	**v-mAP@**	
				B	P	S	V	0.5	0.2	0.5
Mettes et al. [53]	-	-	✓	✓	✓		✓	-	37.4	-
Esc. et al. [74]	-	-	✓	✓			✓	-	45.5	-
Zhang et al. [75]	-	-	✓	✓			✓	30.4	45.5	17.3
Arnab et al. [54]	-	-	✓	✓			✓	-	61.7	35.0
Mettes et al. [55]	-	-	✓		✓		✓	-	41.8	-
Cheron et al. [76]	-	-	✓	✓			✓	-	70.6	38.6
Weinz. et al. [10]	1.1%	52	✓	✓			✓	-	57.1	46.3
Weinz. et al. [10]	2.8%	132	✓	✓			✓	63.8	57.3	46.9
ASL [25]	1.0%	47	✗	✓			✓	64.7	95.3	63.9
ASL [25]	5.0%	235	✗	✓			✓	70.9	96.0	71.9
PSL [137]	S-20%	938	✗	✓			✓	64.9	93.0	65.6
Co-SSD [63]	S-20%	938	✗	✓			✓	65.3	93.7	67.5
Kumar et al. [56]	S-20%	938	✗	✓			✓	69.9	95.7	72.1
Ours (random)	1.0%	28	✗	✓		✓	✓	64.8	95.3	64.6
Ours (random)	6.0%	172	✗	✓		✓	✓	68.2	97.3	67.8
Ours (w/AL)	1.0%	28	✗	✓		✓	✓	65.3	96.1	65.9
Ours (w/AL)	6.0%	172	✗	✓		✓	✓	72.4	97.8	71.2
Ours (Full-sup)	100%	4.6K	✗	✓			✓	75.2	98.8	74.0

Table 6.2: Comparison with state-of-the-art weakly-supervised methods using various annotation type and amount for J-HMDB. We show the percent of annotated frames as $\mathcal{A}\%$ for available methods and their cost as $\mathcal{C}$. $\mathcal{O}$ denotes if the method uses off-the-shelf object detector to generate training annotations. The annotation types used in training is denoted as: M$\rightarrow$ Mask, P$\rightarrow$ Points, S$\rightarrow$ Scribbles, V$\rightarrow$ Video tag. S-30 use semi-supervised training with 30% data. We compare f-mAP and v-mAP at different thresholds.

Method	$\mathcal{A}\%$	$\mathcal{C}$	$\mathcal{O}$	Type				**f-mAP@**	**v-mAP@**	
				M	P	S	V	0.5	0.2	0.5
Zhang et al. [75]	-	-	✓	✓			✓	65.9	77.3	50.8
Weinz. et al. [10]	6%	30	✓	✓			✓	50.7	-	58.5
Weinz. et al. [10]	15%	74	✓	✓			✓	56.5	-	64.0
PSL [137]	S-30%	146	✗	✓			✓	57.4	90.1	57.4
Co-SSD [63]	S-30%	146	✗	✓			✓	60.7	94.3	58.5
Kumar et al. [56]	S-30%	146	✗	✓			✓	64.4	95.4	63.5
Ours (random)	6%	16	✗	✓		✓	✓	56.0	97.1	57.9
Ours (random)	15%	40	✗	✓		✓	✓	67.4	97.3	67.3
Ours (w/AL)	6%	16	✗	✓		✓	✓	58.2	97.2	56.7
Ours (w/AL)	15%	40	✗	✓		✓	✓	70.9	97.6	69.9
Ours (Full-sup)	100%	487	✗	✓			✓	75.8	98.9	74.9

6.3.6.1 Comparison with baselines

We show the performance of our baseline approach in Table 6.1 and 6.2 for UCF-101 and J-HMDB respectively. We compare with another baseline which uses our proposed omni-supervised training approach with random frame selection for annotation instead. Our baseline (**Ours w/AL**) consistently outperforms random selection baseline for both UCF-101 and J-HMDB for equal amount of annotation cost, giving better use of the annotation budget with our training and selection method.

6.3.6.2 *Comparison with prior work*

We compare our approach (**Ours w/AL**) with prior work on UCF-101 and J-HMDB dataset. Compared to most prior works, we use pseudo-labels from interpolation and 3D superpixels instead of external detector. This allows us to use mixed types of annotations, keeping the annotation cost lower than prior approaches for same amount of annotated frames. With our omni-supervised training, our total annotation cost for 6% data (172 hours) is lower than 5% of prior work (235 hours) while performing better in Table 6.1. At 1% annotations we need only 28 man hours of budget to outperform prior works which need 47 man hours [25] and 52 man hours [10]. As shown in Figure 6.4, we are able to obtain comparable performance with fully-supervised approach using 6% training data and using fewer annotation cost (**6% @ 171 man hours**) compared to prior methods (**5% @ 234 man hours, 20% @ 937 man hours**).

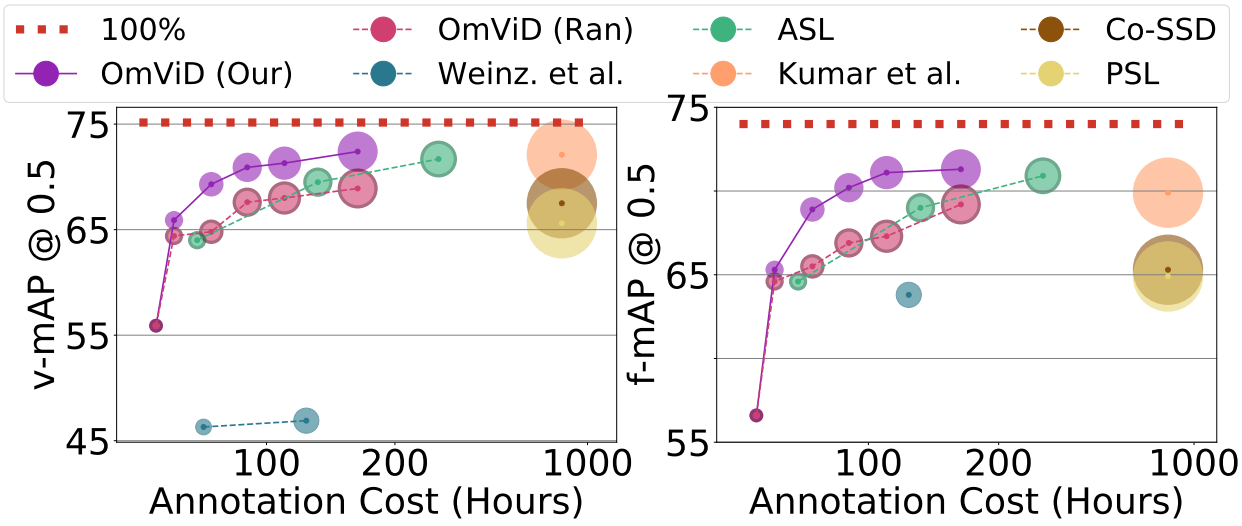

Figure 6.4: Our proposed OmViD approach outperforms prior weakly/semi-supervised video action detection work while using less annotation cost. We show the performance (v-mAP/f-mAP @ 0.5) using various budgets for UCF-101 dataset. The bubble size represents the amount of frames annotated for each method. Our omni-supervised approach is able to annotate more frames at lower cost and perform better, optimizing annotation cost and model performance.

6.3.7 *Ablations*

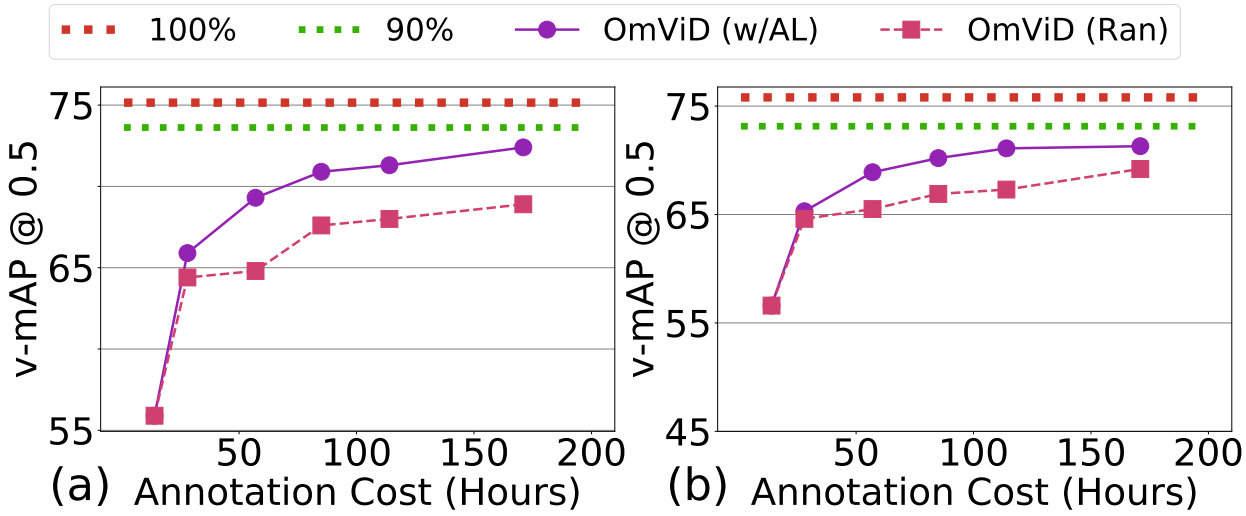

Figure 6.5: Effectiveness of frame selection using our full method (OmViD w/AL) compared to without AL. We use random selection approach and show the impact AL based selection has for selecting useful frames given same annotation budget. We show v-mAP score for UCF-101 (a) and J-HMDB (b).

Without AL selection We evaluate the effectiveness of using proposed AL for increasing training annotations. We use random selection for same amount of annotation in each round for fair comparison, so while the annotation cost stays same the frames selected are at random. We use an initial set of sparse annotations (bounding-box, mask, tags) equally divided among the videos and increase annotations gradually using AL and random method. We show the model performance in figure 6.5. While performance improves for both selection with more rounds, the model performance on random frame selection does not improve as much as proposed AL frame selection for same annotation budget.

Without 3D Superpixel We compare the efficacy of using proposed 3D superpixel to generate pseudo-labels from scribbles with alternative methods for pseudo-label generation. We create

Table 6.3: We compare OmViD training with and without proposed 3D superpixel on J-HMDB dataset. We fit boxes over scribbles instead of proposed 3D superpixel for pseudo-labels in 'wo/Superpixel' method. We report [v-mAP, f-mAP] @ 0.5 IoU scores.

Videos	**Mask**	**Scribbles**	**With Superpixel**		**wo/Superpixel**	
%	%	%	**v-mAP**	**f-mAP**	**v-mAP**	**f-mAP**
30%	1.5%	1.5%	43.4	47.1	41.8	46.6
50%	3.0%	3.0%	56.0	57.9	58.6	61.5
70%	4.5%	4.5%	61.2	65.4	58.9	61.9
90%	6.0%	6.0%	64.9	67.3	64.2	65.4
100%	7.5%	7.5%	67.3	67.4	65.7	66.2
100%	9.0%	9.0%	68.0	68.3	66.6	67.0

pseudo-labels from scribbles by converting it directly to bounding box based on the scribble size as an alternative method. We show the effect of using this alternative pseudo-lable technique on Table 6.3 for J-HMDB dataset. We notice that the bounding box pseudo-labels start falling behind with more annotations added compared to 3D superpixel approach for same annotations. This is primarily due to 3D superpixel generating better annotations from scribbles that is closer to the ground truth.

Annotation type selection We evaluate the importance of having different types of annotations for each round of increment. We use similar setting as baseline but only add scribbles or box/masks in each round instead of proposed mixed increment approach and show the results in Figure 6.6 for UCF-101 and J-HMDB. We notice that while adding only scribbles has lower cost compared to annotating box/masks, the performance gain is also limited. On the other hand, adding box/mask for all those selected frames in a round would increase performance while also increasing annotation cost. Our proposed approach selects mixed annotation types (box/mask, scribbles, tag only) which reduces annotation budget while still improving performance in a comparable way and provides a quantitative method to trade-off between performance and annotation cost.

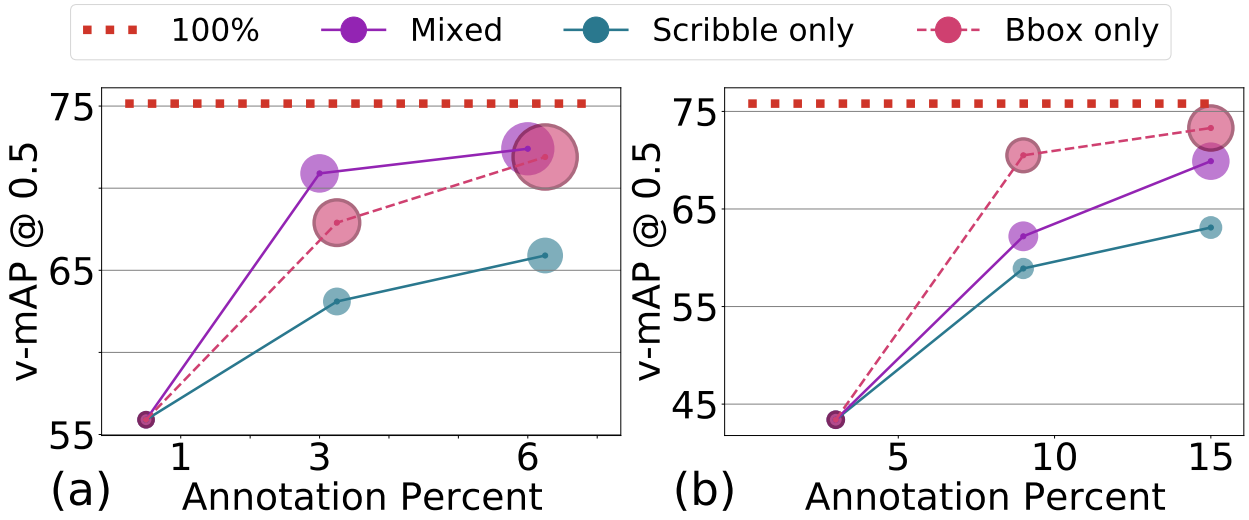

Figure 6.6: We increase only one type of annotation and demonstrate the effectiveness vs annotation cost for doing so. We also compare with the proposed method of increasing all annotation types based on AL ranking. We show v-mAP score for UCF-101 (a) and J-HMDB (b). The bubbles represent the annotation cost hours for each type of annotation increment. For the same percent of annotation, increasing box/mask only incurs large annotation cost hours compared to proposed method with mixed selection. Scribble only increment costs less but also severely underperforms.

6.4 Analysis

Annotation cost reduction Video understanding tasks generally require large annotation budget since each video requires multiple frames annotated. The total annotation budget for UCF-101 with boxes is **4,686** man hours for 2284 training videos from table 6.1 and for J-HMDB with masks is **487** man hours for 666 training videos from table 6.2. This high cost limits dataset scaling compared to image and language domains. Our approach uses 171 man hours (vs 235 hours [25], 938 hours [56]) for UCF-101 and 40 man hours (vs 74 hours [10], 146 hours [56]) for J-HMDB and beats prior weakly-supervised work that use more budget for fewer annotated frames.

Selection approach We provide a quantitative method to select annotations based on the training

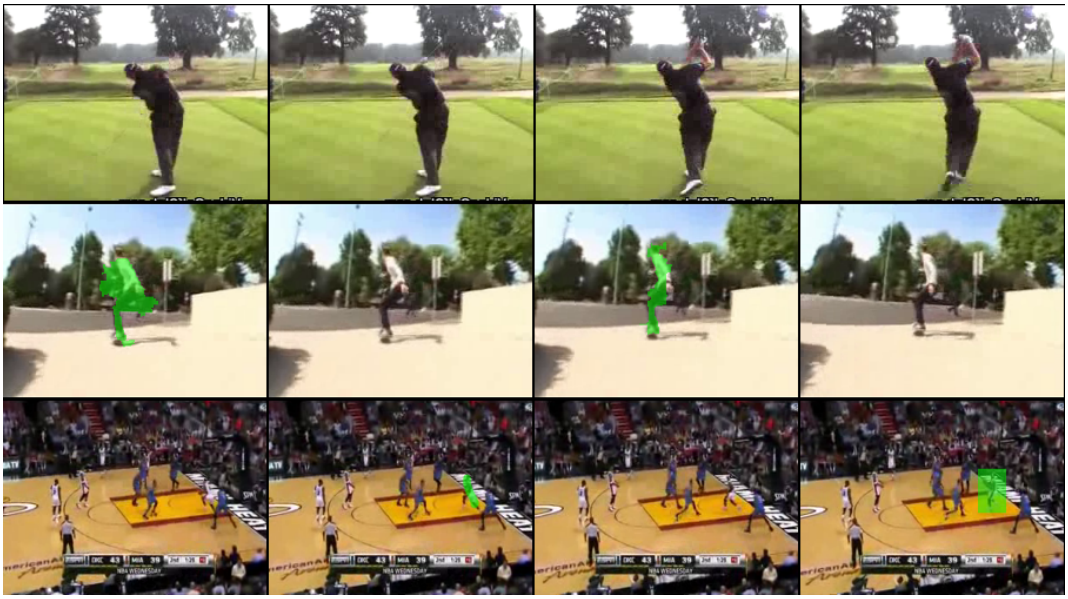

Figure 6.7: Samples with varying annotation types from our method. 1st row: As GolfSwing is easy class, it only needs video level tag to train this sample. 2nd row: Skating is medium hard class, using more scribble annotations. 3rd row: BasketballDunk is very hard class and uses both box and scribble annotation.

needs (video tags $\rightarrow$ scribbles $\rightarrow$ box/mask) as shown in figure 6.7. Using AL selection, we pick samples based on their uncertainty (high to low) for box/mask to tag annotation. We show the selected frame distribution for all 24 classes using the proposed approach for UCF-101 in figure 6.8. We sort the classes from high to low based on their v-mAP score @ 0.5 IoU from left to right. We notice that classes with lower performance often end up needing more box annotations (highest quality and highest cost). These are harder classes for the model to learn action detection and thus demand more accurate annotations. For classes with higher performance on the left (such as SoccerJug., Skating, HorseRiding, and GolfSwing), we notice that more scribble and video tag annotations are selected. Since these classes are easier to learn, the annotation type can be of lower quality and thus also be of lower cost. We observe a mix of box and scribble for classes with average scores in the middle. We demonstrate that AL based selection is a more meaningful use

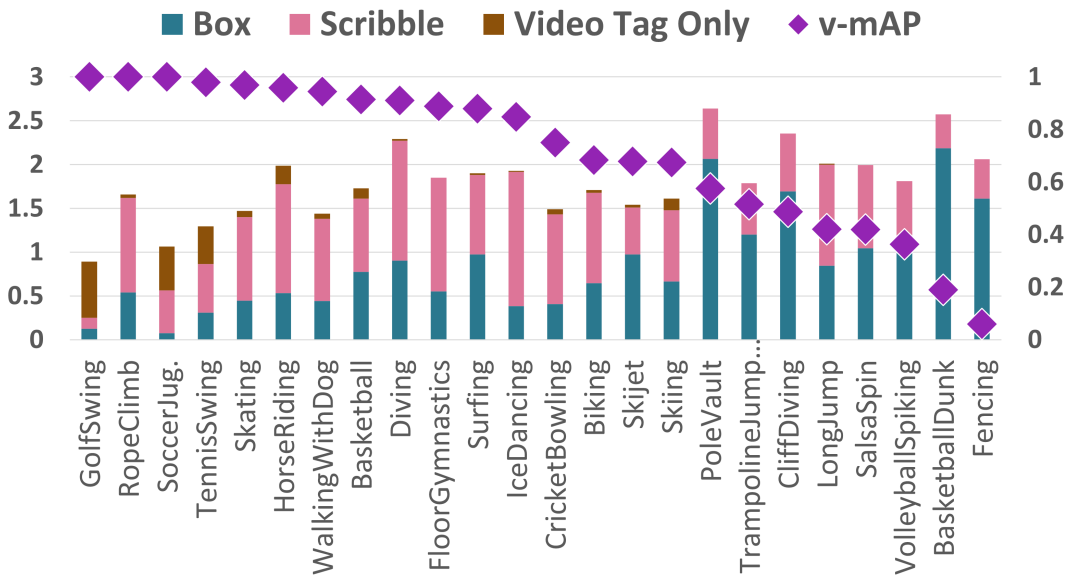

Figure 6.8: Per class annotation distribution for UCF-101. We show the performance of all 24 classes along with the total frames in each class for each type of annotation. The frame count is shown in left axis in thousands **(K)** and the v-mAP @ 0.5 is shown in right axis. We scale the 'video tags only' by 10 to make it more visible.

of the annotation budget as we can extract more out of the same annotation budget compared to random selection. Qualitative visual of proposed approach shown in figure 6.9 and 6.10.

6.5 Summary

In this chapter, we present a unified video action detection model that can be trained with different annotation type (box/mask, scribble, tag) under omni-supervision paradigm. It consists of a learnable 3D superpixel module which is jointly trained with the action detection model and helps in generating pseudo-labels from sparse annotations for effective model training. We also demonstrate that proposed model trains better with AL based frame selection, utilizing the available annotation budget better by selecting appropriate frames for the respective annotation type.

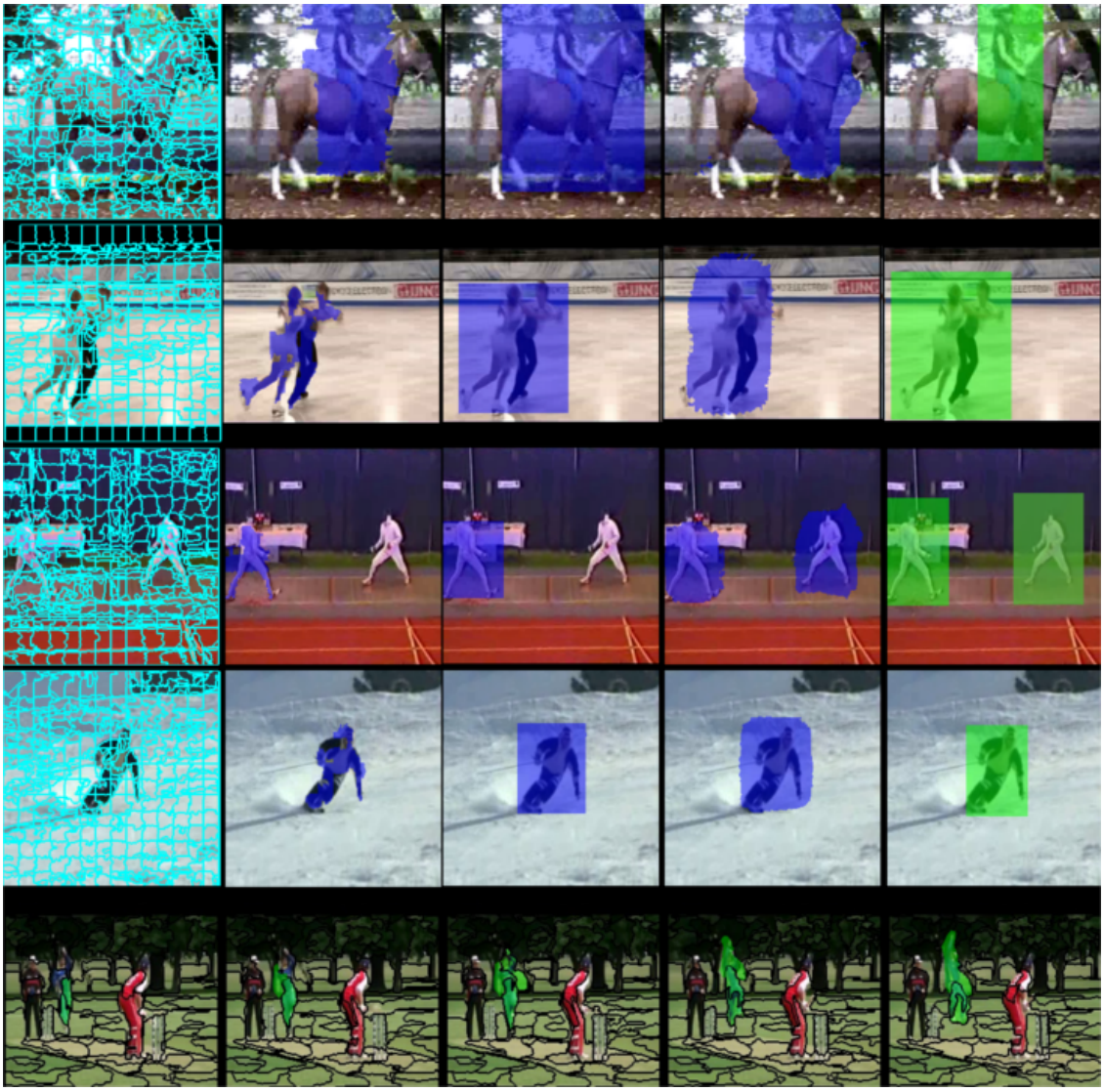

Figure 6.9: Demonstration of our proposed superpixel based pseudo-label generation on UCF-101. We use output of the superpixel (1st col) to get regions that overlays with scribble (2nd col). This is then used to generate pseudo-bounding box for training (3rd col) which trains the network to make action prediction (4th col). The actual ground truth is shown in last column. We show the full 3D superpixel label in last row with the pseudo-label in green highlight.

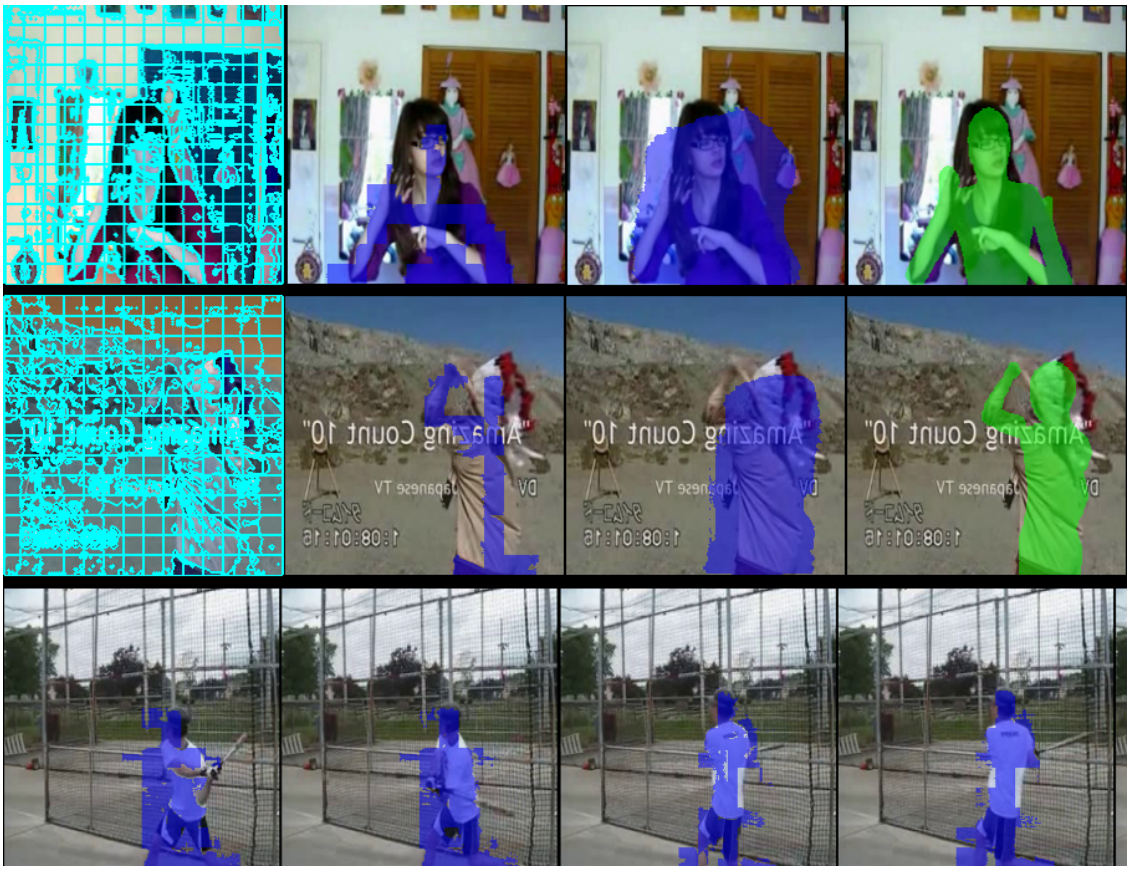

Figure 6.10: Demonstration of our proposed superpixel based pseudo-label generation on J-HMDB. We use output of the superpixel (1st col) to get regions that overlays with scribble (2nd col) and used for network training to predict actions(3rd col). The actual ground truth is shown in last column. We show the full 3D superpixel label in last row with the pseudo-label in blue highlight.

CHAPTER 7: CONCLUSION AND FUTURE WORK

We provide concluding remarks for this dissertation and list the future direction this work can be expanded to.

7.1 Conclusion

In chapter 3, we improve the model efficiency for video action detection in terms of inference speed and model size while improving performance. We propose an end-to-end model for single-shot actor-action detection in videos which does not rely on thousands of proposals for accurate detection. Instead, our proposed model performs a joint pixel-level prediction for actors and actions which removes reliance on external proposal generator reducing network size and computation time. Furthermore, the joint pixel-level prediction approach uses local and global spatio-temporal context to predict correct class for each pixel and improves detection performance. As a result, we are able to perform quicker video actor-action detection on A2D and VidOR datasets with competitive performance.

In chapter 4, we explore improving video understanding from sparse annotations in an effort to reduce the large annotation cost for video dataset. We identify the limitation on video action detection using sparse spatio-temporal annotation. We find that for sparse annotation setting, it is vital to have good representation of the training set such that the model can learn underlying task. We propose an active learning based frame selection approach, which uses video specific properties to improve selection of high utility frames under a given budget. We also provide a simple and effective loss formulation that can be used to train existing video action detection model with sparse annotations. We demonstrate that this proposed active learning selection approach

combined with the novel loss formulation enables reduction in annotation cost for several datasets and video understanding tasks.

In chapter 5, we extend video understanding from sparse annotations further by doing selection at frame as well as video level. We provide a novel hybrid active learning approach which uses cluster information to select high utility videos for annotation and then annotate only few useful frames within those videos. We also improve on the loss formulation to give each pixel individual weight during loss computation based on the spatio-temporal consistency of pseudo-label. Our approach allows comparison across video from different class with different length and complexity. We evaluate the proposed approach on two different action detection datasets demonstrating its effectiveness in learning from limited labels with minimal trade-off on the performance.

Lastly, in chapter 6, we explore the relation between different annotation types, their cost, and their effect on video action detection learning. We provide a omni-supervised training approach for video action detection where several annotation types are used together to train the model. We show how sparse annotation of various types (box, mask, scribble, tag) can be used to train video model effectively while reducing annotation cost. We provide an active learning approach which can help rank samples for different annotation type based on their usefulness. We also provide a learnable 3D superpixel module which is jointly trained with the action detection model, which is used to generate pseudo-labels from sparse anntation of different types. We show that the active learning based ranking allows annotating less informative samples with lower-cost annotation and the 3D superpixel pseudo-label helps train video action detection model for two different video action detection dataset.

7.2 Future work

Video understanding is a challenging topic that needs to be expanded meticulously for integration in everyday tasks. While we have explored and presented several methods for efficient and cost effective video understanding, there are a lot of unexplored aspects which can be studied to expand the findings of this work. Here we outline some of the possible avenues for each of our work.

Chapter 3 provides an efficient model for joint actor-action detection. While it avoids proposal generation and saves computation cost, an interesting direction for future work can be to do instance level detection. Detecting individual instances of actors/actions from same class in a scene is challenging with applications in crowded scene analysis, tracking, and automation. The proposed approach does not handle such conditions as the pixel-wise prediction has no notion of separate instance. Our approach is designed for global-local feature understanding for each pixel to predict the correct actor-action pair, and as such has no loss for identifying correct instance. Some feasible methods for making the approach instance-aware is by associating all foreground detections with unique object entities, which can be achieved using edge or center association in an efficient manner.

Chapter 4 and 5 explores various efficient annotation techniques which reduces redundancy and helps annotate high utility samples for video action understanding task. In chapter 5 we explore frame level sample selection which is limited by not being able to compare and rank different videos against one another. This is due to lack of a fair comparison approach which can account for the model's shortcoming for each video with respect to the underlying task. Videos often have varying action length, number of actors involved, complexity of scene, density of scene with background, and motion of camera as well as subjects. All of these combined makes comparison of videos from one class to another challenging. We provide a method to rank videos across classes and give a solution to this in chapter 6. We also provide methods to use such sparse annotation

effectively for pseudo-label generation as well as loss computation from those pseudo-labels. One common challenge for both frame and video level sample selection was on already sparse data. Existing sparse dataset with 1 frame annotation per second makes using the psuedo-label technique unreliable, giving rise to cold start problem as the initial models are not able to learn meaningful features from the video. Future work can explore a more dynamic pseudo-label generation and loss computation method that uses the motion information to give proper weight to each pseudo-label. Currently, the video selection and ranking approach in chapter 6 suffers when there is small or no video for some classes in initial setup. This is a common cold start problem which needs to be addressed. Since we dont know the labels beforehand, a good solution to this can be to use semi-supervised approach to use partial unlabeled samples in training earlier models, in an attempt to enable feature understanding from underrepresented classes.

Chapter 6 studies relation between various annotation type and their utility for video understanding task and provides an omni-supervised training framework that can use mix of annotation types for video understanding. This is one of the first work of this type for video understanding, and as such has many limitations to get a fair baseline for analysing different annotation type. One limitation is the use of equal proportion of each annotation type in the training set. Ideally, each dataset will have a different optimum proportion of annotation for each type (box, mask, scribble, tag) which gives the best results while using least annotation budget possible. Currently there is no metric to find this optimum proportion of annotations and doing a hyperparameter grid search is not a practical solution as the dataset is evolving. Thus, a good future direction can be to develop metrics which also indicate the contribution of each annotation type in the training process which can help to adjust the proportion of annotation types.

We believe that efficient annotation technique will be valuable to curate and generate high quality video understanding dataset for complex tasks and that the methods presented in this work can be further extended to improve the efficiency for annotation techniques.

LIST OF REFERENCES

[1] C. Xu, S.-H. Hsieh, C. Xiong, and J. J. Corso. Can humans fly? Action understanding with multiple classes of actors. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[2] Hueihan Jhuang, Juergen Gall, Silvia Zuffi, Cordelia Schmid, and Michael J Black. Towards understanding action recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 3192–3199, 2013.

[3] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[4] Ning Xu, Linjie Yang, Yuchen Fan, Jianchao Yang, Dingcheng Yue, Yuchen Liang, Brian Price, Scott Cohen, and Thomas Huang. Youtube-vos: Sequence-to-sequence video object segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 585–601, 2018.

[5] Kevin Duarte, Yogesh Rawat, and Mubarak Shah. Videocapsulenet: A simplified network for action detection. In *Advances in Neural Information Processing Systems*, pages 7610–7619, 2018.

[6] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[7] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, pages 1050–1059. PMLR, 2016.

[8] Hamed H Aghdam, Abel Gonzalez-Garcia, Joost van de Weijer, and Antonio M López. Active learning for deep detection neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3672–3680, 2019.

[9] Ho Kei Cheng, Yu-Wing Tai, and Chi-Keung Tang. Rethinking space-time networks with improved memory coverage for efficient video object segmentation. In *Advances in Neural Information Processing Systems*, 2021.

[10] Philippe Weinzaepfel, Xavier Martin, and Cordelia Schmid. Human action localization with sparse spatial supervision. *arXiv preprint arXiv:1605.05197*, 2016.

[11] Madeline Chantry Schiappa, Naman Biyani, Prudvi Kamtam, Shruti Vyas, Hamid Palangi, Vibhav Vineet, and Yogesh S Rawat. A large-scale robustness analysis of video action recognition models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14698–14708, 2023.

[12] Madeline Chantry Schiappa, Shruti Vyas, Hamid Palangi, Yogesh S Rawat, and Vibhav Vineet. Robustness analysis of video-language models against visual and language perturbations. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

[13] Joshua Gleason, Carlos D Castillo, and Rama Chellappa. Real-time detection of activities in untrimmed videos. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision Workshops*, pages 117–125, 2020.

[14] Mamshad Nayeem Rizve, Ugur Demir, Praveen Tirupattur, Aayush Jung Rana, Kevin Duarte, Ishan R Dave, Yogesh S Rawat, and Mubarak Shah. Gabriella: An online system for real-time activity detection in untrimmed security videos. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 4237–4244. IEEE, 2021.

[15] Markus Schön, Michael Buchholz, and Klaus Dietmayer. Mgnet: Monocular geometric scene understanding for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15804–15815, 2021.

[16] Di Feng, Christian Haase-Schütz, Lars Rosenbaum, Heinz Hertlein, Claudius Glaeser, Fabian Timm, Werner Wiesbeck, and Klaus Dietmayer. Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges. *IEEE Transactions on Intelligent Transportation Systems*, 22(3):1341–1360, 2020.

[17] Alex X Lee, Richard Zhang, Frederik Ebert, Pieter Abbeel, Chelsea Finn, and Sergey Levine. Stochastic adversarial video prediction. *arXiv preprint arXiv:1804.01523*, 2018.

[18] Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. *arXiv preprint arXiv:1605.07157*, 2016.

[19] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6546–6555, 2018.

[20] Rui Hou, Chen Chen, and Mubarak Shah. An end-to-end 3d convolutional neural network for action detection and segmentation in videos. *arXiv preprint arXiv:1712.01111*, 2017.

[21] Kang Dang, Chunluan Zhou, Zhigang Tu, Michael Hoy, Justin Dauwels, and Junsong Yuan. Actor-action semantic segmentation with region masks. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2018.

[22] Jingwei Ji, Shyamal Buch, Alvaro Soto, and Juan Carlos Niebles. End-to-end joint semantic segmentation of actors and actions in video. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 702–717, 2018.

[23] Rui Hou, Chen Chen, and Mubarak Shah. Tube convolutional neural network (t-cnn) for action detection in videos. In *IEEE International Conference on Computer Vision*, 2017.

[24] Xitong Yang, Xiaodong Yang, Ming-Yu Liu, Fanyi Xiao, Larry S Davis, and Jan Kautz. Step: Spatio-temporal progressive learning for video action detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 264–272, 2019.

[25] Aayush J. Rana and Yogesh S. Rawat. Are all frames equal? active sparse labeling for video action detection. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, 2022.

[26] Yubo Zhang, Pavel Tokmakov, Martial Hebert, and Cordelia Schmid. A structured model for action detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9975–9984, 2019.

[27] Jiajun Tang, Jin Xia, Xinzhi Mu, Bo Pang, and Cewu Lu. Asynchronous interaction aggregation for action detection. In *European Conference on Computer Vision*, pages 71–87. Springer, 2020.

[28] Chunhui Gu, Chen Sun, Sudheendra Vijayanarasimhan, Caroline Pantofaru, David A Ross, George Toderici, Yeqing Li, Susanna Ricco, Rahul Sukthankar, Cordelia Schmid, et al. Ava: A video dataset of spatio-temporally localized atomic visual actions. corr abs/1705.08421 (2017). *arXiv preprint arXiv:1705.08421*, 2017.

[29] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5188–5197, 2019.

[30] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[31] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[32] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[33] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. *arXiv preprint*, 1612, 2016.

[34] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[35] Georgia Gkioxari, Ross Girshick, Piotr Dollár, and Kaiming He. Detecting and recognizing human-object interactions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8359–8367, 2018.

[36] Zhenheng Yang, Jiyang Gao, and Ram Nevatia. Spatio-temporal action detection with cascade proposal and location anticipation. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2017.

[37] Xiaojiang Peng and Cordelia Schmid. Multi-region two-stream r-cnn for action detection. In *European conference on computer vision*, pages 744–759. Springer, 2016.

[38] Dong Li, Zhaofan Qiu, Qi Dai, Ting Yao, and Tao Mei. Recurrent tubelet proposal and recognition networks for action detection. In *Proceedings of the European conference on computer vision (ECCV)*, pages 303–318, 2018.

[39] Zhaofan Qiu, Ting Yao, and Tao Mei. Learning spatio-temporal representation with pseudo-3d residual networks. In *proceedings of the IEEE International Conference on Computer Vision*, pages 5533–5541, 2017.

[40] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 305–321, 2018.

[41] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Computer Vision (ICCV), 2015 IEEE International Conference on*, pages 4489–4497. IEEE, 2015.

[42] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014.

[43] Shruti Vyas, Yogesh S Rawat, and Mubarak Shah. Multi-view action recognition using cross-view video prediction. In *Proceedings of the European Conference on Computer Vision*, 2020.

[44] Rohit Girdhar, Joao Carreira, Carl Doersch, and Andrew Zisserman. Video action transformer network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 244–253, 2019.

[45] Jiaojiao Zhao, Xinyu Li, Chunhui Liu, Shuai Bing, Hao Chen, Cees GM Snoek, and Joseph Tighe. Tuber: Tube-transformer for action detection. *arXiv preprint arXiv:2104.00969*, 2021.

[46] Chenliang Xu and Jason J Corso. Actor-action semantic segmentation with grouping process models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3083–3092, 2016.

[47] Yan Yan, Chenliang Xu, Dawen Cai, and Jason J Corso. Weakly supervised actor-action segmentation via robust multi-task ranking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1298–1307, 2017.

[48] Vicky Kalogeiton, Philippe Weinzaepfel, Vittorio Ferrari, and Cordelia Schmid. Joint learning of object and action detectors. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4163–4172, 2017.

[49] Kirill Gavrilyuk, Amir Ghodrati, Zhenyang Li, and Cees GM Snoek. Actor and action video segmentation from a sentence. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5958–5966, 2018.

[50] Bruce McIntosh, Kevin Duarte, Yogesh S Rawat, and Mubarak Shah. Visual-textual capsule routing for text-based video segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9942–9951, 2020.

[51] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[52] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[53] Pascal Mettes, Cees GM Snoek, and Shih-Fu Chang. Localizing actions from video labels and pseudo-annotations. *arXiv preprint arXiv:1707.09143*, 2017.

[54] Anurag Arnab, Chen Sun, Arsha Nagrani, and Cordelia Schmid. Uncertainty-aware weakly supervised action detection from untrimmed videos. In *European Conference on Computer Vision*, pages 751–768. Springer, 2020.

[55] Pascal Mettes and Cees GM Snoek. Pointly-supervised action localization. *International Journal of Computer Vision*, 127(3):263–281, 2019.

[56] Akash Kumar and Yogesh Singh Rawat. End-to-end semi-supervised learning for video action detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14700–14710, 2022.

[57] Ross Girshick, Ilija Radosavovic, Georgia Gkioxari, Piotr Dollár, and Kaiming He. Detectron. `https://github.com/facebookresearch/detectron`, 2018.

[58] Anna Khoreva, Rodrigo Benenson, Jan Hosang, Matthias Hein, and Bernt Schiele. Simple does it: Weakly supervised instance and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 876–885, 2017.

[59] Gurkirt Singh, Suman Saha, Michael Sapienza, Philip HS Torr, and Fabio Cuzzolin. Online real-time multiple spatiotemporal action localisation and prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3637–3646, 2017.

[60] Zhongzheng Ren, Raymond Yeh, and Alexander Schwing. Not all unlabeled data are equal: Learning to weight data in semi-supervised learning. *Advances in Neural Information Processing Systems*, 33:21786–21797, 2020.

[61] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *arXiv preprint arXiv:2001.07685*, 2020.

[62] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. *Advances in neural information processing systems*, 32, 2019.

[63] Jisoo Jeong, Seungeui Lee, Jeesoo Kim, and Nojun Kwak. Consistency-based semi-supervised learning for object detection. *Advances in neural information processing systems*, 32, 2019.

[64] Ishan Rajendrakumar Dave, Mamshad Nayeem Rizve, Chen Chen, and Mubarak Shah. Timebalance: Temporally-invariant and temporally-distinctive video representations for semi-supervised action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2341–2352, 2023.

[65] Mengde Xu, Zheng Zhang, Han Hu, Jianfeng Wang, Lijuan Wang, Fangyun Wei, Xiang Bai, and Zicheng Liu. End-to-end semi-supervised object detection with soft teacher. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3060–3069, 2021.

[66] Yandong Li, Di Huang, Danfeng Qin, Liqiang Wang, and Boqing Gong. Improving object detection with selective self-supervised self-training. In *European Conference on Computer Vision*, pages 589–607. Springer, 2020.

[67] Mamshad Nayeem Rizve, Kevin Duarte, Yogesh S Rawat, and Mubarak Shah. In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning. In *International Conference on Learning Representations*, 2021.

[68] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5297–5307, 2016.

[69] Mamshad Nayeem Rizve, Gaurav Mittal, Ye Yu, Matthew Hall, Sandra Sajeev, Mubarak Shah, and Mei Chen. Pivotal: Prior-driven supervision for weakly-supervised temporal action localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22992–23002, 2023.

[70] Julia Peyre, Josef Sivic, Ivan Laptev, and Cordelia Schmid. Weakly-supervised learning of visual relations. In *Proceedings of the ieee international conference on computer vision*, pages 5179–5188, 2017.

[71] Mamshad Nayeem Rizve, Navid Kardan, Salman Khan, Fahad Shahbaz Khan, and Mubarak Shah. Openldn: Learning to discover novel classes for open-world semi-supervised learning. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXI*, pages 382–401. Springer, 2022.

[72] Mamshad Nayeem Rizve, Navid Kardan, and Mubarak Shah. Towards realistic semi-supervised learning. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXI*, pages 437–455. Springer, 2022.

[73] Huy V Vo, Oriane Siméoni, Spyros Gidaris, Andrei Bursuc, Patrick Pérez, and Jean Ponce. Active learning strategies for weakly-supervised object detection. In *European Conference on Computer Vision*, pages 211–230. Springer, 2022.

[74] Victor Escorcia, Cuong D Dao, Mihir Jain, Bernard Ghanem, and Cees Snoek. Guess where? actor-supervision for spatiotemporal action localization. *Computer Vision and Image Understanding*, 192:102886, 2020.

[75] Shiwei Zhang, Lin Song, Changxin Gao, and Nong Sang. Glnet: Global local network for weakly supervised action localization. *IEEE Transactions on Multimedia*, 22(10):2610–2622, 2019.

[76] Guilhem Chéron, Jean-Baptiste Alayrac, Ivan Laptev, and Cordelia Schmid. A flexible model for training action localization with varying levels of supervision. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 950–961, 2018.

[77] Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid. Learning to track for spatio-temporal action localization. In *Proceedings of the IEEE international conference on computer vision*, pages 3164–3172, 2015.

[78] Burr Settles. Active learning literature survey. 2009.

[79] Xin Li and Yuhong Guo. Adaptive active learning for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 859–866, 2013.

[80] Keze Wang, Dongyu Zhang, Ya Li, Ruimao Zhang, and Liang Lin. Cost-effective active learning for deep image classification. *IEEE Transactions on Circuits and Systems for Video Technology*, 27(12):2591–2600, 2016.

[81] Ajay J Joshi, Fatih Porikli, and Nikolaos Papanikolopoulos. Multi-class active learning for image classification. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2372–2379. IEEE, 2009.

[82] Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Brij B Gupta, Xiaojiang Chen, and Xin Wang. A survey of deep active learning. *ACM Computing Surveys (CSUR)*, 54(9):1–40, 2021.

[83] Guang Zhao, Edward Dougherty, Byung-Jun Yoon, Francis Alexander, and Xiaoning Qian. Uncertainty-aware active learning for optimal bayesian classifier. In *International Conference on Learning Representations (ICLR 2021)*, 2021.

[84] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *International Conference on Machine Learning*, pages 1183–1192. PMLR, 2017.

[85] William H Beluch, Tim Genewein, Andreas Nürnberger, and Jan M Köhler. The power of ensembles for active learning in image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9368–9377, 2018.

[86] Ashish Kapoor, Kristen Grauman, Raquel Urtasun, and Trevor Darrell. Active learning with gaussian processes for object categorization. In *2007 IEEE 11th international conference on computer vision*, pages 1–8. IEEE, 2007.

[87] Zimo Liu, Jingya Wang, Shaogang Gong, Huchuan Lu, and Dacheng Tao. Deep reinforcement active learning for human-in-the-loop person re-identification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6122–6131, 2019.

[88] Alex Holub, Pietro Perona, and Michael C Burl. Entropy-based active learning for object recognition. In *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8. IEEE, 2008.

[89] Andreas Kirsch, Joost Van Amersfoort, and Yarin Gal. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. *Advances in neural information processing systems*, 32, 2019.

[90] Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*, 2011.

[91] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*, 2017.

[92] Robert Pinsler, Jonathan Gordon, Eric Nalisnick, and José Miguel Hernández-Lobato. Bayesian batch active learning as sparse subset approximation. *Advances in Neural Information Processing Systems*, 32, 2019.

[93] Carl Vondrick and Deva Ramanan. Video annotation and tracking with active learning. *Advances in Neural Information Processing Systems*, 24:28–36, 2011.

[94] Javad Zolfaghari Bengar, Abel Gonzalez-Garcia, Gabriel Villalonga, Bogdan Raducanu, Hamed Habibi Aghdam, Mikhail Mozerov, Antonio M López, and Joost van de Weijer. Temporal coherence for active learning in videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[95] Fabian Caba Heilbron, Joon-Young Lee, Hailin Jin, and Bernard Ghanem. What do i annotate next? an empirical study of active learning for action localization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 199–216, 2018.

[96] Bishan Yang, Jian-Tao Sun, Tengjiao Wang, and Zheng Chen. Effective multi-label active learning for text classification. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 917–926, 2009.

[97] Ye Zhang, Matthew Lease, and Byron Wallace. Active discriminative text representation learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[98] Ameya Prabhu, Charles Dognin, and Maneesh Singh. Sampling bias in deep active classification: An empirical study. *arXiv preprint arXiv:1909.09389*, 2019.

[99] Dilek Hakkani-Tür, Giuseppe Riccardi, and Allen Gorin. Active learning for automatic speech recognition. In *2002 IEEE international conference on acoustics, speech, and signal processing*, volume 4, pages IV–3904. IEEE, 2002.

[100] Soumya Roy, Asim Unmesh, and Vinay P Namboodiri. Deep active learning for object detection. In *BMVC*, page 91, 2018.

[101] Jiwoong Choi, Ismail Elezi, Hyuk-Jae Lee, Clement Farabet, and Jose M Alvarez. Active learning for deep object detection via probabilistic modeling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10264–10273, 2021.

[102] Tianning Yuan, Fang Wan, Mengying Fu, Jianzhuang Liu, Songcen Xu, Xiangyang Ji, and Qixiang Ye. Multiple instance active learning for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5330–5339, 2021.

[103] Alireza Fathi, Maria Florina Balcan, Xiaofeng Ren, and James M Rehg. Combining self training and active learning for video segmentation. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2011.

[104] Zhongzheng Ren, Zhiding Yu, Xiaodong Yang, Ming-Yu Liu, Alexander G Schwing, and Jan Kautz. Ufo^2: A unified framework towards omni-supervised object detection. In *European Conference on Computer Vision*, pages 288–313. Springer, 2020.

[105] Pei Wang, Zhaowei Cai, Hao Yang, Gurumurthy Swaminathan, Nuno Vasconcelos, Bernt Schiele, and Stefano Soatto. Omni-detr: Omni-supervised object detection with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9367–9376, 2022.

[106] Jia Xu, Alexander G Schwing, and Raquel Urtasun. Learning to segment under various forms of weak supervision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3781–3790, 2015.

[107] Liangyu Chen, Tong Yang, Xiangyu Zhang, Wei Zhang, and Jian Sun. Points as queries: Weakly semi-supervised object detection by points. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8823–8832, 2021.

[108] Zhenheng Yang, Dhruv Mahajan, Deepti Ghadiyaram, Ram Nevatia, and Vignesh Ramanathan. Activity driven weakly supervised object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2917–2926, 2019.

[109] Zhongzheng Ren and Yong Jae Lee. Cross-domain self-supervised multi-task feature learning using synthetic imagery. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 762–771, 2018.

[110] Jasper Uijlings, Stefan Popov, and Vittorio Ferrari. Revisiting knowledge transfer for training object class detectors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1101–1110, 2018.

[111] Bruce D Lucas, Takeo Kanade, et al. *An iterative image registration technique with an application to stereo vision*, volume 81. Vancouver, 1981.

[112] Vicky Kalogeiton, Philippe Weinzaepfel, Vittorio Ferrari, and Cordelia Schmid. Action tubelet detector for spatio-temporal action localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4405–4413, 2017.

[113] Yaxiong Wang, Yunchao Wei, Xueming Qian, Li Zhu, and Yi Yang. Ainet: Association implantation for superpixel segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7078–7087, 2021.

[114] Fengting Yang, Qian Sun, Hailin Jin, and Zihan Zhou. Superpixel segmentation with fully convolutional networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13964–13973, 2020.

[115] Aayush J. Rana and Yogesh S. Rawat. We don't need thousand proposals: Single shot actor-action detection in videos. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2960–2969, January 2021.

[116] Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Robert Fergus. Deconvolutional networks. In *proceedings of the IEEE International Conference on Computer Vision*, volume 10, pages 2528–2535, 2010.

[117] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[118] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 801–818, 2018.

[119] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2117–2125, 2017.

[120] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. *arXiv preprint arXiv:1901.02446*, 2019.

[121] Carole H Sudre, Wenqi Li, Tom Vercauteren, Sebastien Ourselin, and M Jorge Cardoso. Generalised dice overlap as a deep learning loss function for highly unbalanced segmen-

tations. In *Deep learning in medical image analysis and multimodal learning for clinical decision support*, pages 240–248. Springer, 2017.

[122] François Chollet et al. Keras, 2015.

[123] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[124] Xindi Shang, Donglin Di, Junbin Xiao, Yu Cao, Xun Yang, and Tat-Seng Chua. Annotating objects and relations in user-generated videos. In *Proceedings of the 2019 on International Conference on Multimedia Retrieval*, pages 279–287. ACM, 2019.

[125] Jie Chen, Zhiheng Li, Jiebo Luo, and Chenliang Xu. Learning a weakly-supervised video actor-action segmentation model with a wise selection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9901–9911, 2020.

[126] Prateek Jain and Ashish Kapoor. Active learning for large multi-class problems. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 762–769. IEEE, 2009.

[127] Xin Li and Yuhong Guo. Adaptive active learning for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 859–866, 2013.

[128] Yi Yang, Zhigang Ma, Feiping Nie, Xiaojun Chang, and Alexander G Hauptmann. Multi-class active learning by uncertainty sampling with diversity maximization. *International Journal of Computer Vision*, 113(2):113–127, 2015.

[129] Geoffrey E Hinton, Sara Sabour, and Nicholas Frosst. Matrix capsules with em routing. In *International conference on learning representations*, 2018.

[130] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[131] Gunnar A Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *European Conference on Computer Vision*, pages 510–526. Springer, 2016.

[132] Yu-Lun Liu, Yi-Tung Liao, Yen-Yu Lin, and Yung-Yu Chuang. Deep video frame interpolation using cyclic frame generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8794–8802, 2019.

[133] Georgia Gkioxari and Jitendra Malik. Finding action tubes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 759–768, 2015.

[134] Chunhui Gu, Chen Sun, David A Ross, Carl Vondrick, Caroline Pantofaru, Yeqing Li, Sudheendra Vijayanarasimhan, George Toderici, Susanna Ricco, Rahul Sukthankar, et al. Ava: A video dataset of spatio-temporally localized atomic visual actions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6047–6056, 2018.

[135] Aayush J Rana and Yogesh S Rawat. Hybrid active learning via deep clustering for video action detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18867–18877, 2023.

[136] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149, 2018.

[137] Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896, 2013.

[138] Jingshan Xu, Chuanwei Zhou, Zhen Cui, Chunyan Xu, Yuge Huang, Pengcheng Shen, Shaoxin Li, and Jian Yang. Scribble-supervised semantic segmentation inference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15354–15363, 2021.

[139] Di Lin, Jifeng Dai, Jiaya Jia, Kaiming He, and Jian Sun. Scribblesup: Scribble-supervised convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3159–3167, 2016.

[140] Alejandro Pardo, Mengmeng Xu, Ali Thabet, Pablo Arbeláez, and Bernard Ghanem. Baod: budget-aware object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1247–1256, 2021.

[141] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[142] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575, 2015.